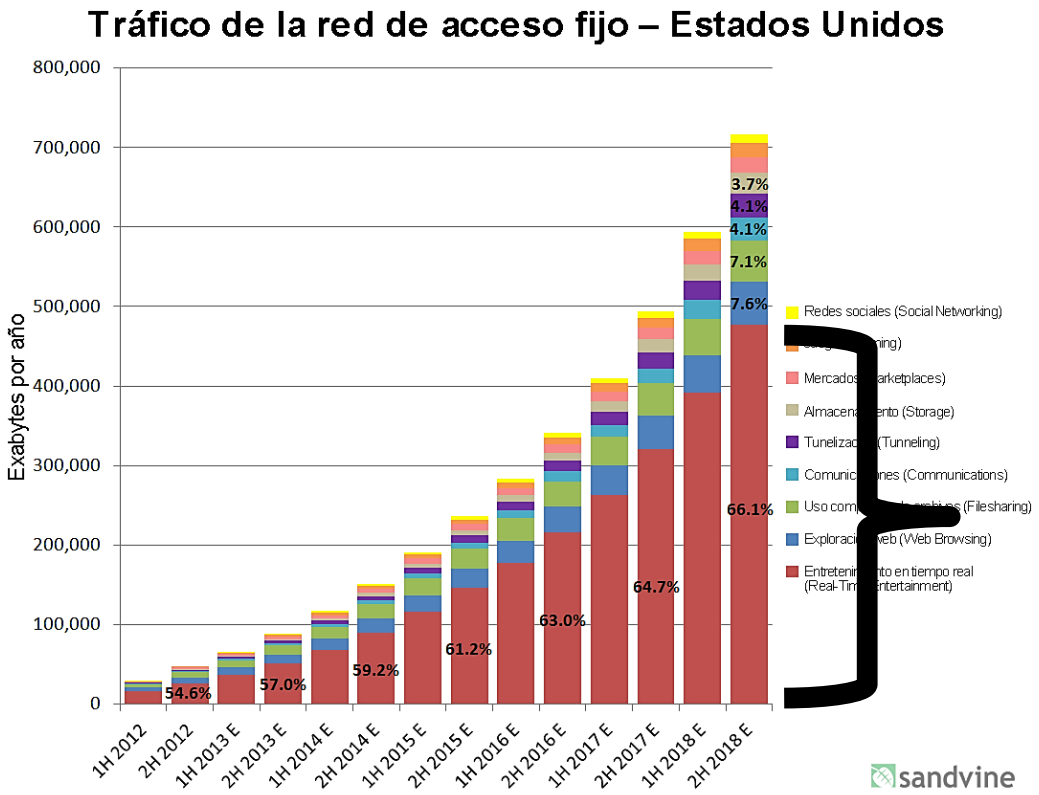
Transportando multimedia

Departamento de INGENIERIA TELEMATICA
www.it.uc3m.es

Tipos de comunicación multimedia

- **Según forma de comunicación: Interactiva / no-interactiva**
  - **No-interactivo: predomina *one-way vídeo***
- **No interactiva, según quién provee**
  - **IPTV: el proveedor que conecta 'la última milla'**
    - ✓ **Ej: Movistar, Vodafone, Jazztel, etc.**
  - **OTT (*Over The Top*): provee otro distinto del proveedor *last-mile***
    - ✓ **Ej: YouTube, RTVE, Netflix …**
- **No interactiva, Según forma de distribución**
  - **Broadcast**
  - **Switched digital video ~ multicast**
  - **On-demand ~ unicast**



Principalmente one-way vídeo, OTT

¿Qué tecnología?

Tecnología para one-way video, versión #1

Servidor de video

```
/* nos centramos en el transporte */
file = open  (fileName , O_RDONLY);
socketTCP = sock (…);
/* establece sesion TCP */

while (datosEnv < tamanoFich)
{
  read (file, memor, TAM_BLOQ);
  write (socketTCP, memor, TAM_BLOQ);
  /* actualiza datosEnv */
}
```

Envío agresivo: tengo los datos, los intento enviar lo más rápido que se pueda (porque están grabados)

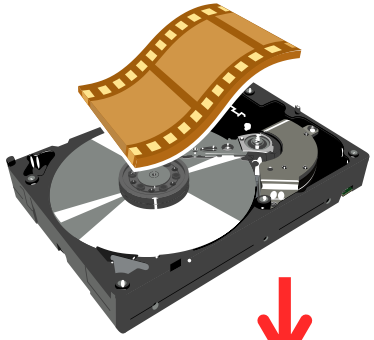
Cliente de video

```
/* nos centramos en el transporte */
socketTCP = sock (…);
/* establece sesion TCP */
/* configura dispositivo de video */
videoDisp = open ('/dev/…' );

while (datosRec< tamanoFich)
{
   read (socketTCP, mem, TAM_BLOQ);
   /* actualiza datosRec */
   write (videoDisp, mem, TAM_BLOQ);
}
```

#1 = transmite video completo, reproduce bloque de datos nada más recibirlo

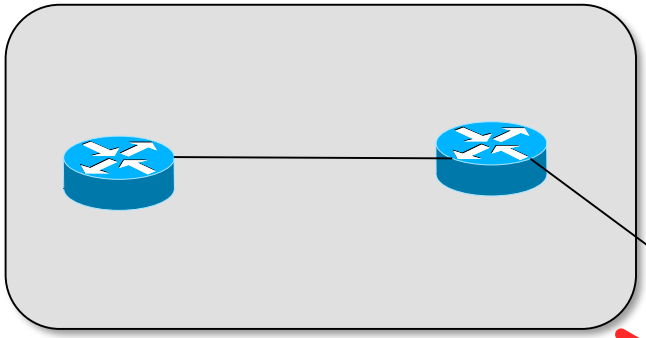
¿Funciona #1?: Control de flujo



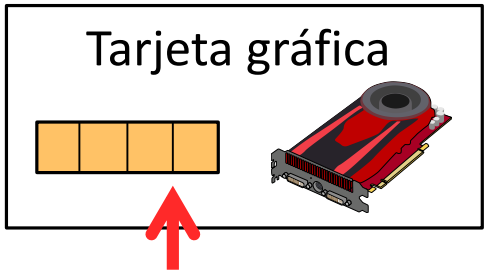
Aplicación

```
while (datosEnv < tamanoFich)
{
   read (file, memor, TAM_BLOQ);
   write (socketTCP, memor, TAM_BLOQ);
   /* actualiza datosEnv */
}
```

Red

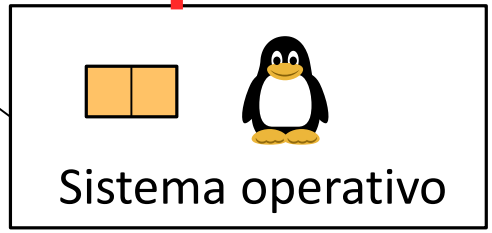


TCP



Aplicación

```
while (datosRec< tamanoFich)
{
   read (socketTCP, mem, TAM_BLOQ);
   /* actualiza datosRec */
   write (videoDisp, mem, TAM_BLOQ);
}
```



Progressive download, streaming, DASH

- Si en el ejemplo anterior añadimos el uso de HTTP (encima de TCP), tendríamos
  - Mecanismo para identificar contenido (url)
  - Un servidor universal y sencillo (cualquier servidor HTTP), con posibilidad de autenticación, etc.
    - ✓ Servidor no mantiene más estado por cliente que el necesario para enviar un fichero
  - *Progressive download*: mecanismo que utiliza HTTP para gestionar acceso a todo el video, fácil de implementar
- *Streaming*: uso de un protocolo específico para comunicar cliente y servidor multimedia
  - Comandos como 'Ir a HH:MM:SS', 'Fast forward x2', …
  - El servidor entiende el formato, mantiene estado por cliente
  - Puede soportar adaptación de calidad
  - Servidor sólo manda lo que el cliente necesita
    - ✓ Ir a HH:MM:SS no necesita que el cliente descargue TODO lo anterior
- DASH: Dynamic Adaptive Streaming over HTTP (DASH)
  - Similar a progressive download, pero los contenidos están en trozos pequeños (pocos segundos)
    - ✓ Puede haber varias versiones del mismo contenido con distintas calidades
  - Estandariza cómo comunicar información de 'trozos' del contenido desde el servidor al cliente
  - La selección del trozo deseado la realiza la lógica del cliente
  - (respecto a progressive download) Facilita saltos, cambios de calidad, etc.
  - Sigue manteniendo el modelo de servidor simple de progressive download

Ejemplo de manifiesto DASH

```
<?xml version="1.0" encoding="utf-8"?> <MPD xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="urn:mpeg:DASH:schema:MPD:2011"
xsi:schemaLocation="urn:mpeg:DASH:schema:MPD:2011 DASH-MPD.xsd"
xmlns:yt="http://youtube.com/yt/2012/10/10" profiles="urn:mpeg:dash:profile:isoff-on-demand:2011" type="static" minBufferTime="PT1.500S" mediaPresentationDuration="PT135.743S">
<Period> <AdaptationSet mimeType="audio/webm" subsegmentAlignment="true"> <Representation id="171" codecs="vorbis" audioSamplingRate="44100" startWithSAP="1" bandwidth="129553">
<AudioChannelConfiguration
schemeIdUri="urn:mpeg:dash:23003:3:audio_channel_configuration:2011" value="2" />
<BaseURL>feelings_vp9-20130806-171.webm</BaseURL> <SegmentBase indexRange="4452-4686"
indexRangeExact="true"> <Initialization range="0-4451" /> </SegmentBase> </Representation>
<Representation id="172" codecs="vorbis" audioSamplingRate="44100" startWithSAP="1"
bandwidth="188041"> <AudioChannelConfiguration
schemeIdUri="urn:mpeg:dash:23003:3:audio_channel_configuration:2011" value="2" />
<BaseURL>feelings_vp9-20130806-172.webm</BaseURL> <SegmentBase indexRange="3995-4229"
indexRangeExact="true"> <Initialization range="0-3994" /> </SegmentBase> </Representation>
</AdaptationSet> <AdaptationSet mimeType="video/webm" subsegmentAlignment="true">
<Representation id="242" codecs="vp9" width="426" height="240" startWithSAP="1"
bandwidth="490208"> <BaseURL>feelings_vp9-20130806-242.webm</BaseURL> <SegmentBase
indexRange="234-682" indexRangeExact="true"> <Initialization range="0-233" /> </SegmentBase>
</Representation>
<Representation id="243" codecs="vp9" width="640" height="360" startWithSAP="1"
bandwidth="927221"> <BaseURL>feelings_vp9-20130806-243.webm</BaseURL> <SegmentBase
indexRange="235-683" indexRangeExact="true"> <Initialization range="0-234" /> </SegmentBase>
</Representation>
```

Calidad en servicios multimedia

Documentos ITU-T relacionados

- G.1000-G.1999: Multimedia Quality of Service and performance – Generic and user-related aspects
  - G.1000: Communications Quality of Service: A framework and definitions
  - G.1010: End-user multimedia QoS categories
  - G.1011: Reference guide to quality of experience assessment methodologies
  - G.1020: Performance parameter definitions for quality of speech and other voiceband applications utilizing IP networks
  - G.1021: Buffer models for development of client performance metrics
  - G.1022: Buffer Models for Media Streams on TCP Transport
  - G.1028: End-to-end quality of service for voice over 4G mobile networks
  - G.1029: Voice service diagnosis framework
  - G.1030: Estimating end-to-end performance in IP networks for data applications
  - G.1031: QoE factors in web-browsing
  - G.1040: Network contribution to transaction time
  - G.1050: Network model for evaluating multimedia transmission performance over Internet Protocol
  - G.1070: Opinion model for video-telephony applications
  - G.1071: Opinion model for network planning of video and audio streaming applications
  - G.1080: Quality of experience requirements for IPTV services
  - G.1081: Performance monitoring points for IPTV
  - G.1082: Measurement-based methods for improving the robustness of IPTV performance
  - G.1091: Quality of Experience requirements for telepresence services

¿cumple progressive download parámetros de calidad?

ITU-T G.1010: 'Categorías de calidad de servicio para los usuarios de extremo de servicios multimedios', 2001

Medium	Application	Degree of symmetry	Typical data rates	Key performance parameters and target values			
				One-way delay	Delay variation	Information loss (Note 2)	Other
Audio	Conversational voice	Two-way	4-64 kbit/s	<150 ms preferred (Note 1) <400 ms limit (Note 1)	< 1 ms	< 3% packet loss ratio (PLR)	
Audio	Voice messaging	Primarily one-way	4-32 kbit/s	< 1 s for playback < 2 s for record	< 1 ms	< 3% PLR	
Audio	High quality streaming audio	Primarily one-way	16-128 kbit/s (Note 3)	< 10 s	<< 1 ms	< 1% PLR	
Video	Videophone	Two-way	16-384 kbit/s	< 150 ms preferred (Note 4) <400 ms limit		< 1% PLR	Lip-synch: < 80 ms
Video	One-way	One-way	16-384 kbit/s	< 10 s		< 1% PLR	

NOTE 1 – Assumes adequate echo control.

NOTE 2 – Exact values depend on specific codec, but assumes use of a packet loss concealment algorithm to minimise effect of packet loss.

NOTE 3 – Quality is very dependent on codec type and bit-rate.

NOTE 4 – These values are to be considered as long-term target values which may not be met by current technology.

Jitter

En la representación

¿cumple progressive download parámetros de calidad?

ITU-T G.1010: 'Categorías de calidad de servicio para los usuarios de extremo de servicios multimedios', 2001

Medium	Application	Degree of symmetry	Typical data rates	Key performance parameters and target values			
				One-way delay	Delay variation	Information loss (Note 2)	Other
Audio	Conversational voice	Two-way	4-64 kbit/s	<150 ms preferred (Note 1) <400 ms limit (Note 1)	< 1 ms	< 3% packet loss ratio (PLR)	
Audio	Voice messaging	Primarily one-way	4-32 kbit/s	< 1 s for playback < 2 s for record	< 1 ms	< 3% PLR	
Audio	High quality streaming audio	Primarily one-way	16-128 kbit/s (Note 3)	< 10 s	<< 1 ms	< 1% PLR	
Video	Videophone	Two-way	16-384 kbit/s	< 150 ms preferred (Note 4) <400 ms limit		< 1% PLR	Lip-synch: < 80 ms
Video	One-way	One-way	16-384 kbit/s	< 10 s		< 1% PLR	

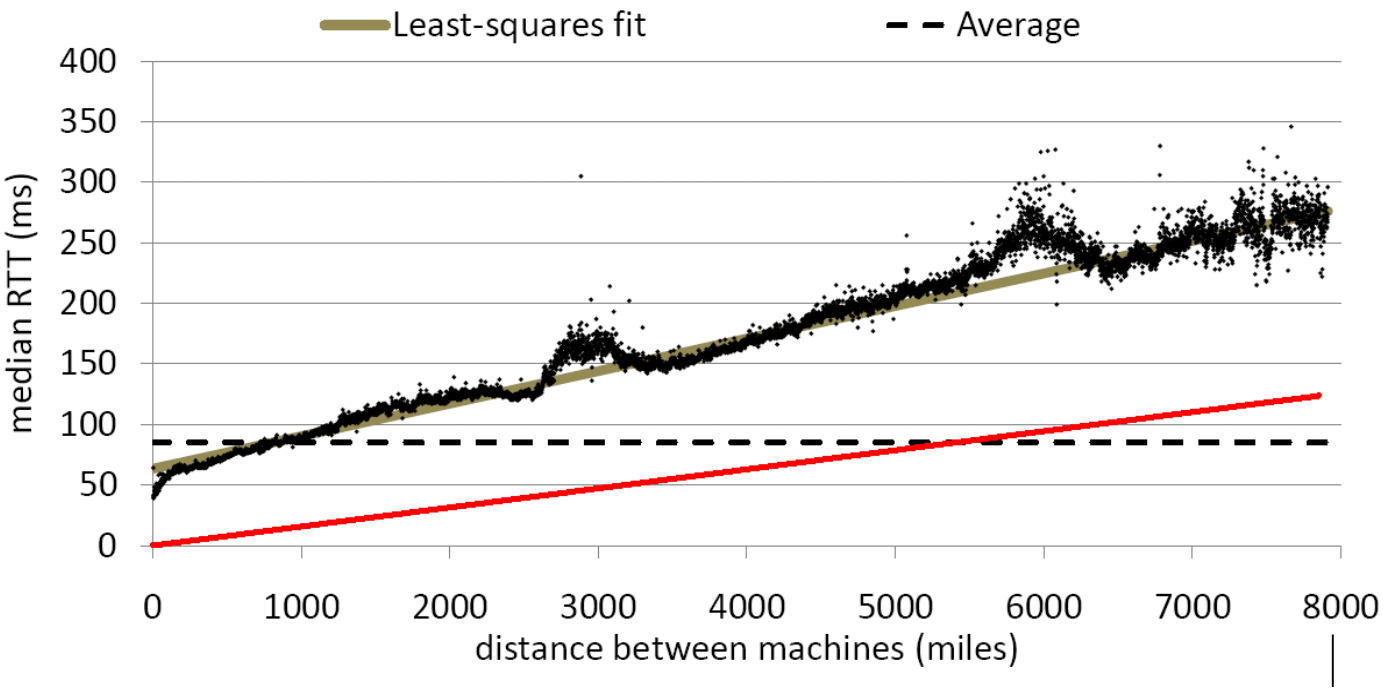
NOTE 1 – Assumes adequate echo control.

NOTE 2 – Exact values depend on specific codec, but assumes use of a packet loss concealment algorithm to minimise effect of packet loss.

NOTE 3 – Quality is very dependent on codec type and bit-rate.

NOTE 4 – These values are to be considered as long-term target values which may not be met by current technology.

RTT, network delay



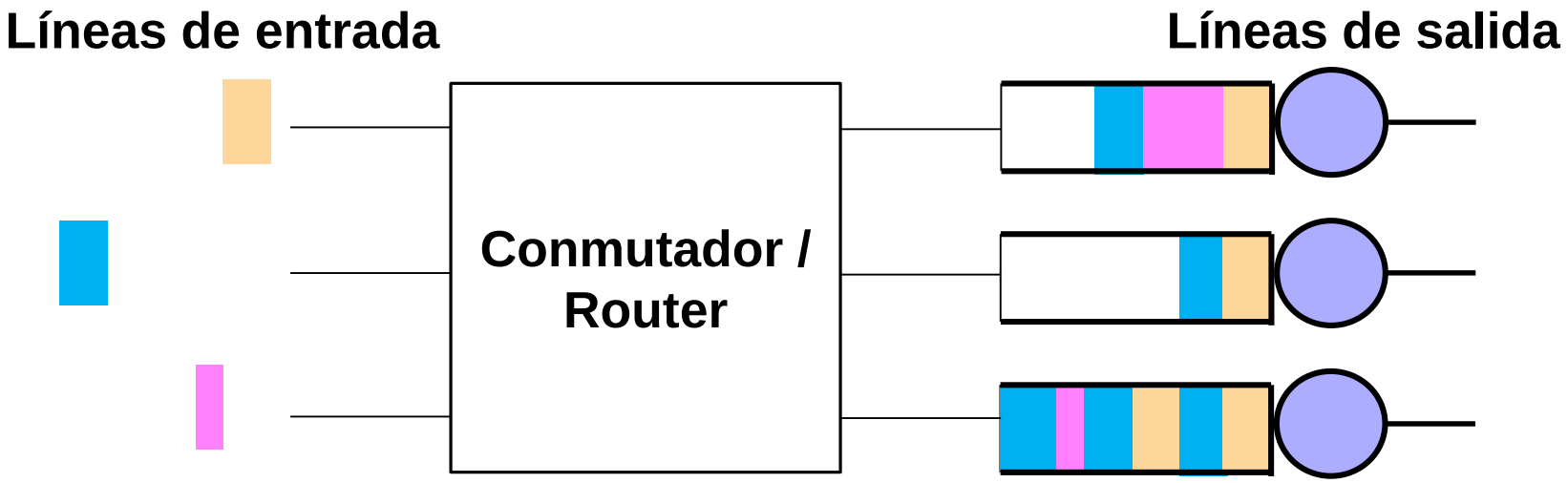
RTT (=ida y vuelta) en función de distancia
Medido para jugadores de Halo-3.
En rojo, retardo de transmisión (RTT) mínimo
Agarwal, Lorch [Sigcomm09]

12874 km

¿cumple progressive download parámetros de calidad?

Medium	Application	Degree of symmetry	Typical data rates	Key performance parameters and target values			
				One-way delay	Delay variation	Information loss (Note 2)	Other
Video	One-way	One-way	16-384 kbit/s	< 10 s		< 1% PLR	

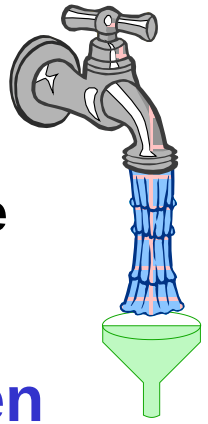
Pérdidas y retardos en la red



- ¿Causas de pérdida de paquetes?
  - Congestión
  - Otras causas: routing
  - ¿Cómo son las pérdidas si utilizo TCP?
- ¿Causas de retardos de paquetes?
  - ¿Puedo cuantificar el retardo máximo 'one-way'?
  - ¿Cómo es el retardo si utilizo TCP?
- ¿Cómo varía retardo / pérdida de paquetes con la distancia?

¿ancho de banda en *progressive download*?

- TCP no pierde paquetes, pero la reproducción puede ser inaceptable si la red no tiene **ancho de banda disponible** suficiente
- Ancho de banda disponible depende de la congestión en el enlace más cargado
  - Menor probabilidad de encontrar congestión si se atraviesan pocos enlaces
    - ✓ Cuando el contenido está cerca del destino
- Ancho de banda disponible puede cambiar en la escala de tiempo de la reproducción del contenido
  - Solución, acumular datos mientras hay ancho de banda suficiente: buffering
- Si no hay ancho de banda suficiente, reducir tasa de envío del contenido, reduciendo calidad



¿cumple *progressive download* parámetros de calidad?

Medium	Application	Degree of symmetry	Typical data rates	Key performance parameters and target values			
				One-way delay	Delay variation	Information loss (Note 2)	Other
Video	One-way	One-way	16-384 kbit/s	< 10 s		< 1% PLR	

- Information loss = 0 en TCP
- Depende de disponer de ancho de banda suficiente

DASH en acción

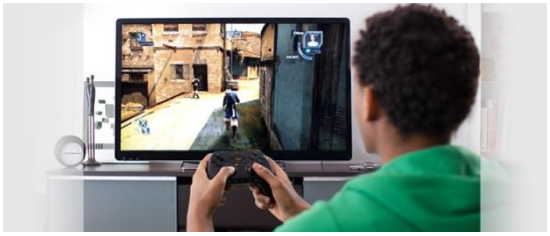
Multicast

- Ni TCP (ni HTTP) permite uso de multicast

~~Transporte de video one-way~~

Transporte de comunicaciones interactivas

Comunicaciones Interactivas



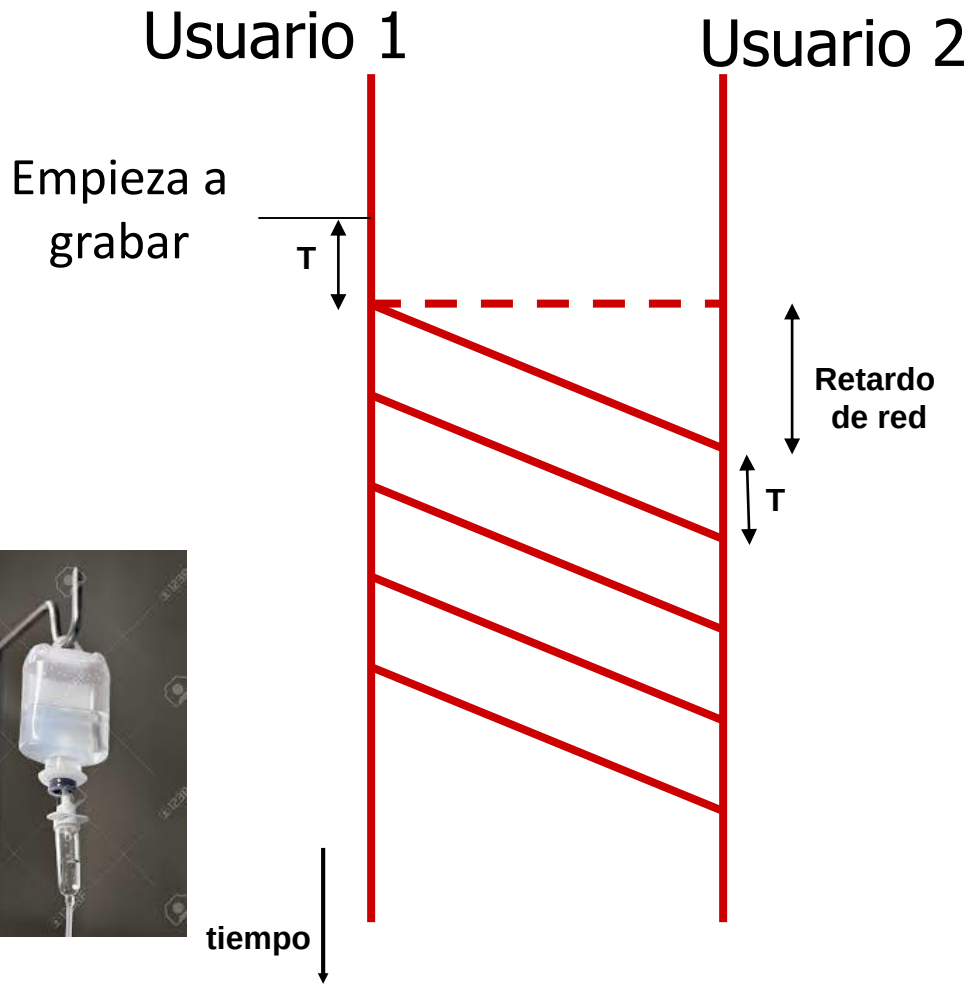
onLIVE

Cloud gaming **requiere respuesta en <120 ms.**

¡El contenido se genera en tiempo real ("sobre la marcha")!

Comunicación interactiva, versión #1

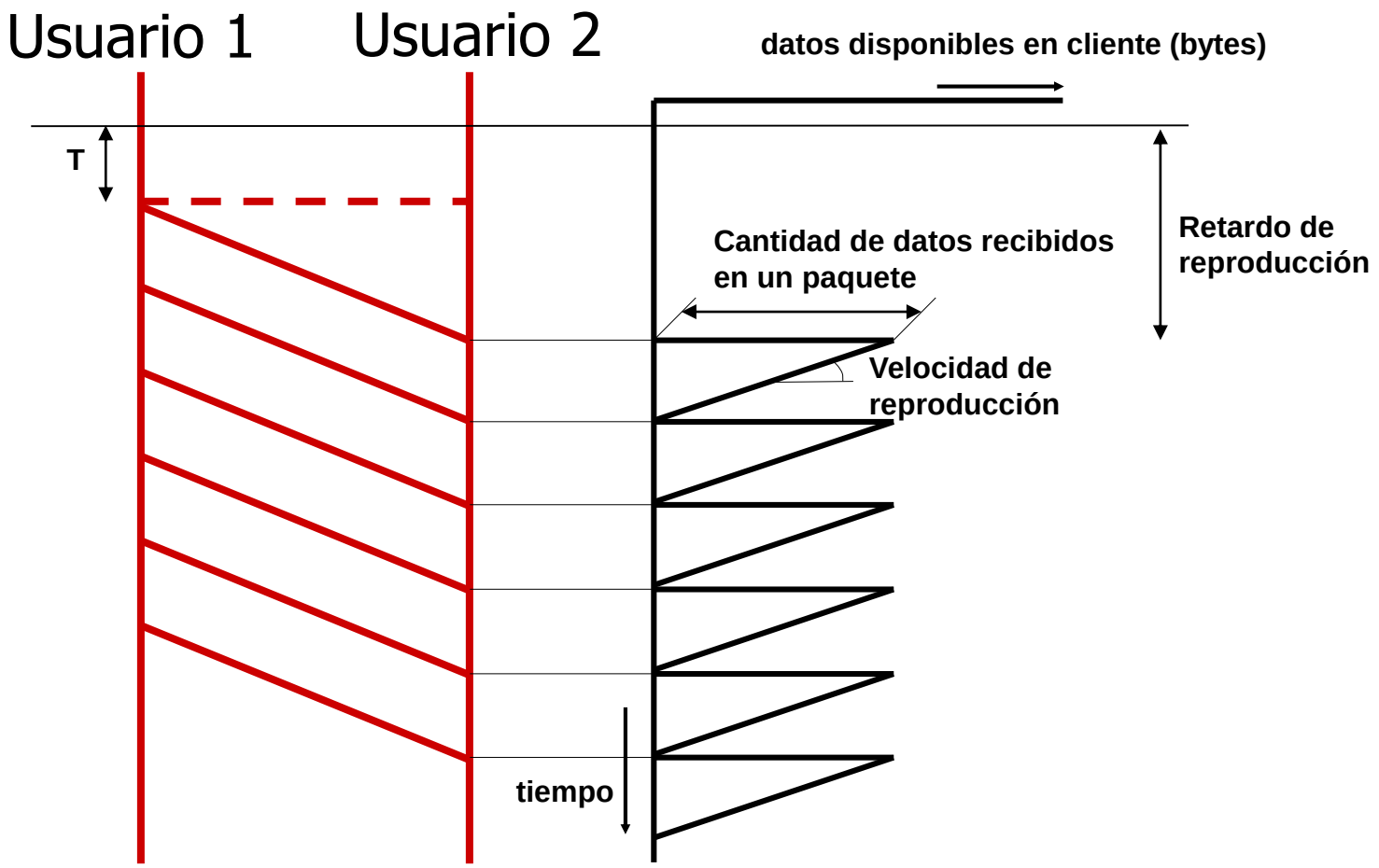
- **Ejemplo típico:**
  - **Audio generado a 8000 muestras por segundo, 1 byte por muestra, mono**
  - **T=20 ms (160 bytes)**
  - **Retardo de codificación (*codec delay*): T**
    - ✓ **Depende de tamaño de paquete (aunque puede ser mayor)**
    - ✓ **Siempre presente**



Comunicación interactiva, versión #1

#1 = reproduce bloque de datos nada más recibirlo

- **Análisis de datos disponibles en la memoria del cliente**



#1: ¿cumple parámetros de calidad?

#1 = reproduce bloque de datos nada más recibirlo

ITU G.1010: 'Categorías de calidad de servicio para los usuarios de extremo de servicios multimedios', 2001

Medium	Application	Degree of symmetry	Typical data rates	Key performance parameters and target values			
				One-way delay	Delay variation	Information loss (Note 2)	Other
Audio	Conversational voice	Two-way	4-64 kbit/s	<150 ms preferred (Note 1) <400 ms limit (Note 1)	< 1 ms 	< 3% packet loss ratio (PLR)	
Audio	Voice messaging	Primarily one-way	4-32 kbit/s	< 1 s for playback < 2 s for record	< 1 ms	< 3% PLR	
Audio	High quality streaming audio	Primarily one-way	16-128 kbit/s (Note 3)	< 10 s	<< 1 ms	< 1% PLR	
Video	Videophone	Two-way	16-384 kbit/s	< 150 ms preferred (Note 4) <400 ms limit		< 1% PLR	Lip-synch: < 80 ms
Video	One-way	One-way	16-384 kbit/s	< 10 s		< 1% PLR	

NOTE 1 – Assumes adequate echo control.

NOTE 2 – Exact values depend on specific codec, but assumes use of a packet loss concealment algorithm to minimise effect of packet loss.

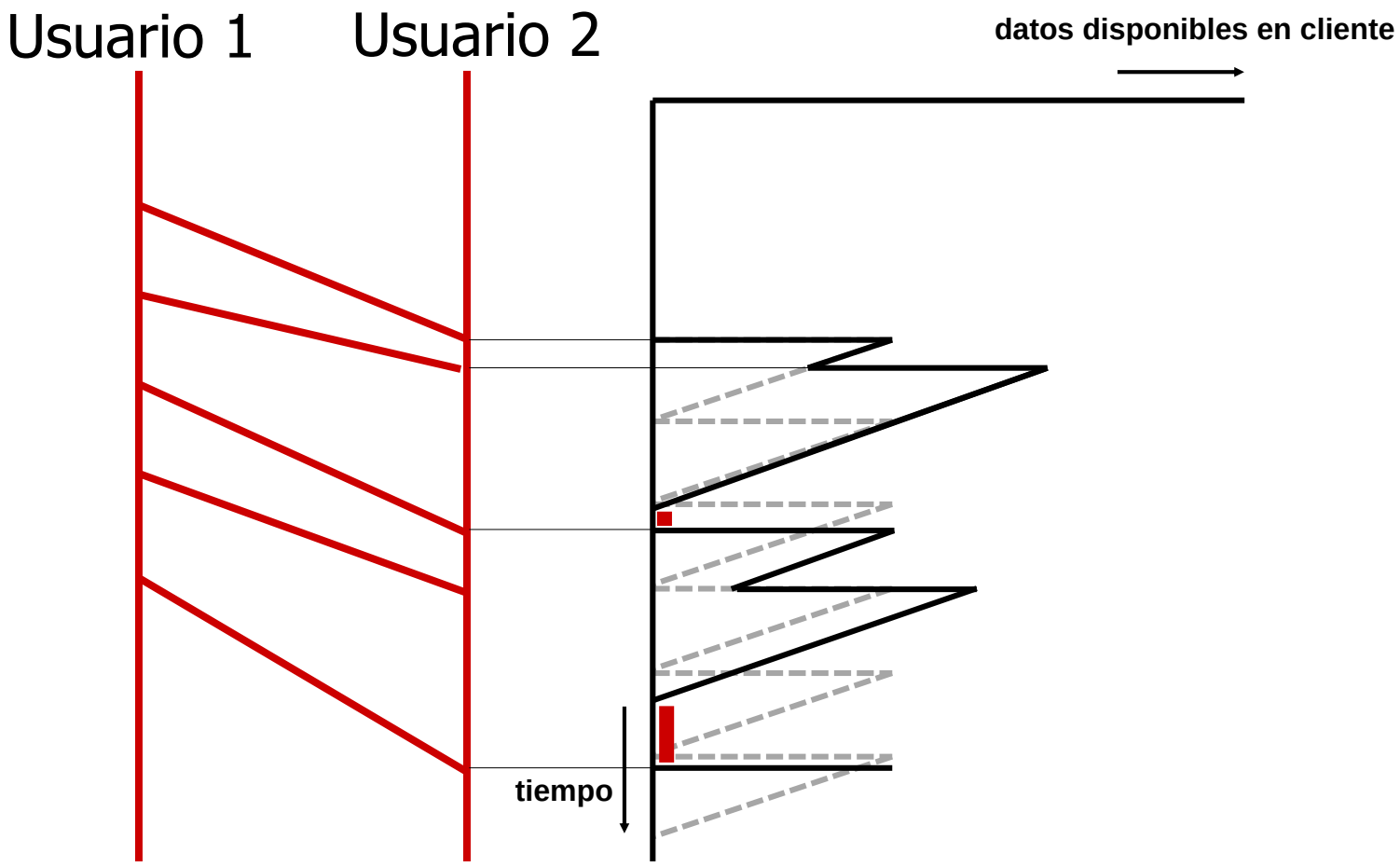
NOTE 3 – Quality is very dependent on codec type and bit-rate.

NOTE 4 – These values are to be considered as long-term target values which may not be met by current technology.

Comunicación interactiva, versión #1

#1 = reproduce bloque de datos nada más recibirlo

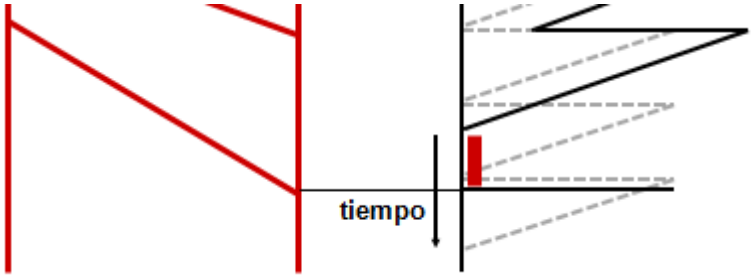
- **Comunicación con retardos variables en la red**



#1: ¿cumple parámetros de calidad?

#1 = reproduce bloque de datos nada más recibirlo

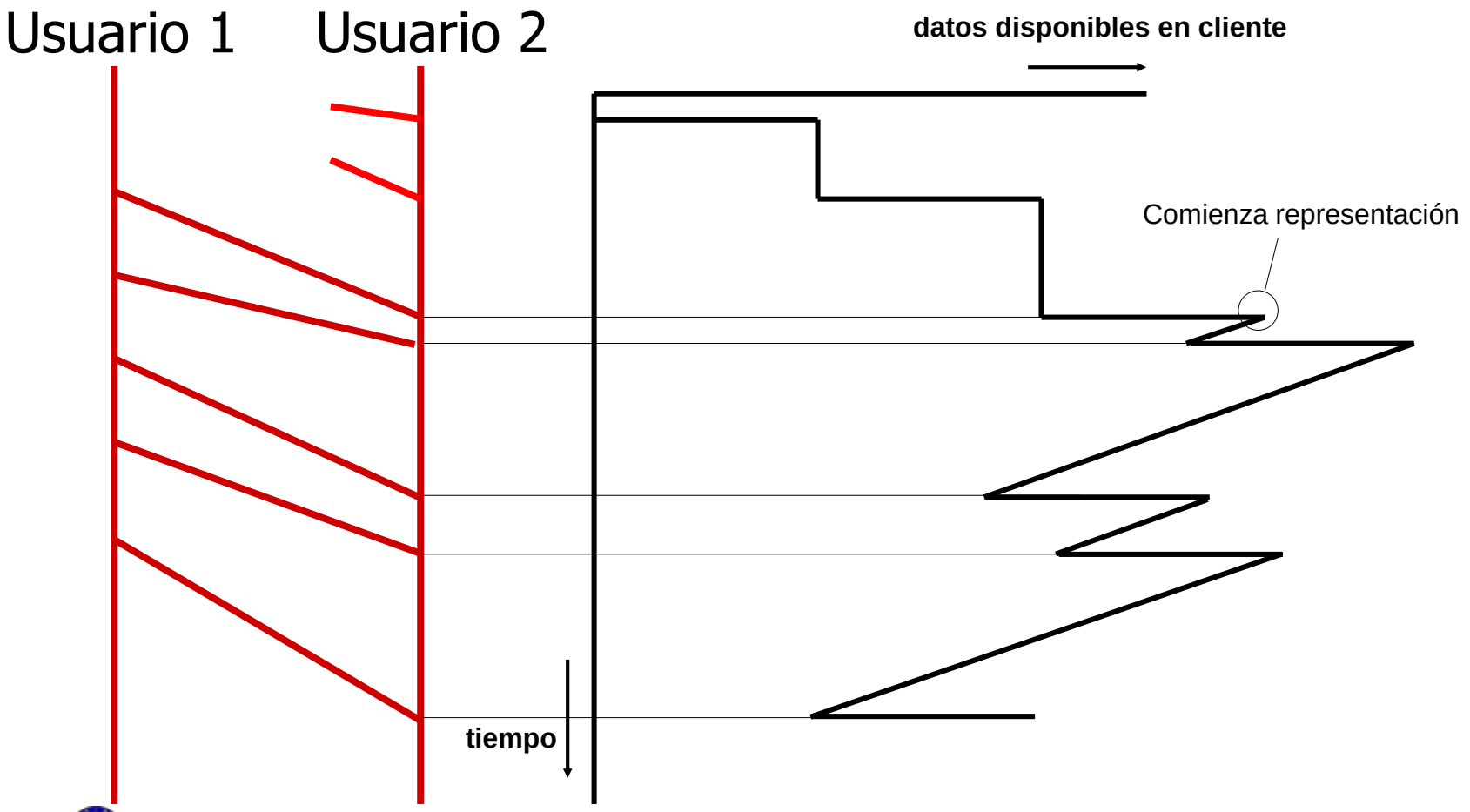
- **Delay variation <1 ms**



- **Necesitamos Versión #2: BUFFERING**

Comunicación interactiva, versión #2

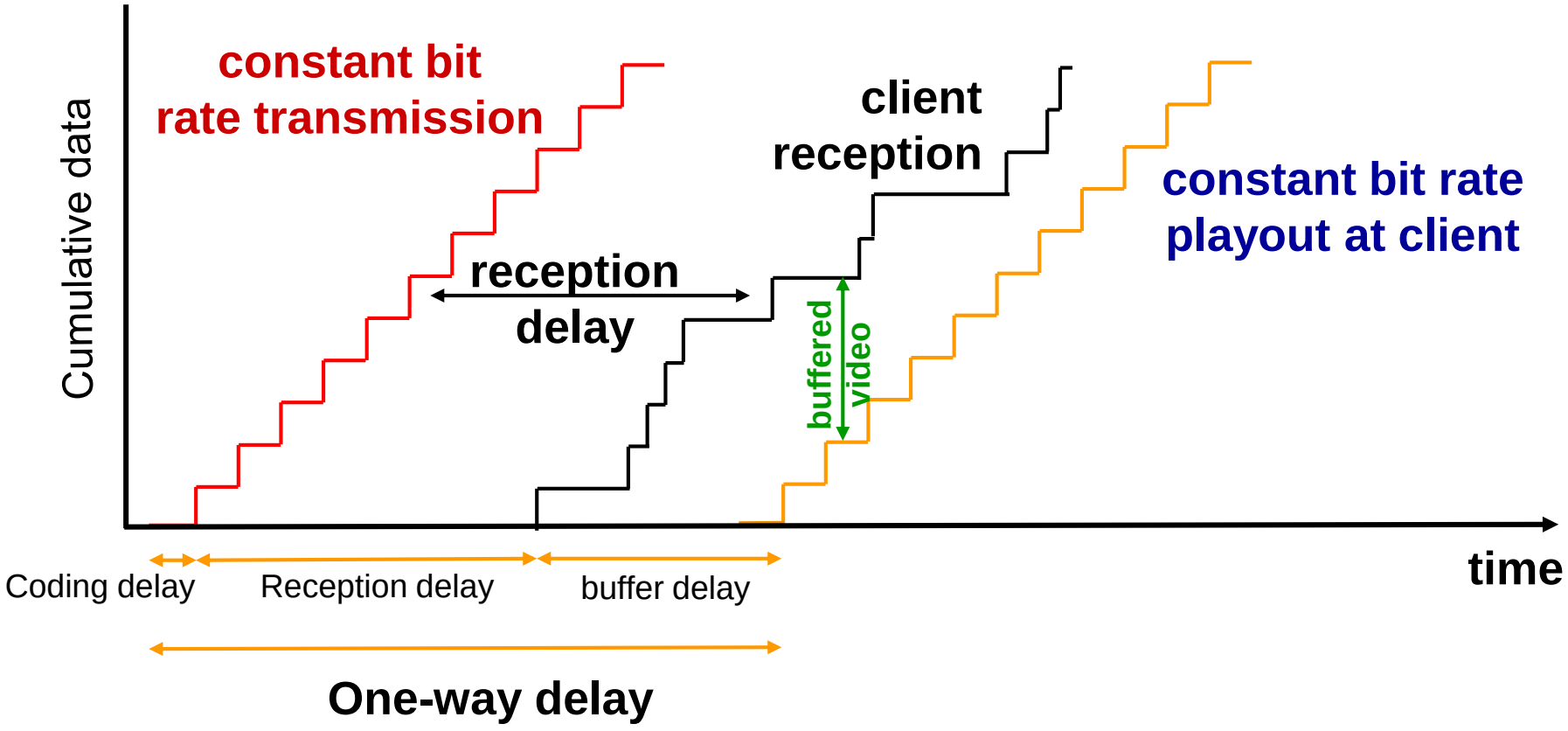
#2= buffering, acumula algunos datos antes de comenzar reproducción



Comunicación interactiva, versión #2

#2= buffering, acumula algunos datos antes de comenzar reproducción

- Otra forma de verlo



Computer Networking: A Top Down Approach, 6th ed,, J. Kurose, K. Ross

#1: ¿cumple parámetros de calidad?

#1 = reproduce bloque de datos nada más recibirlo

ITU G.1010: 'Categorías de calidad de servicio para los usuarios de extremo de servicios multimedios', 2001

Medium	Application	Degree of symmetry	Typical data rates	Key performance parameters and target values			
				One-way delay	Delay variation	Information loss (Note 2)	Other
Audio	Conversational voice	Two-way	4-64 kbit/s	<150 ms preferred (Note 1) <400 ms limit (Note 1)	< 1 ms	< 3% packet loss ratio (PLR)	
Audio	Voice messaging	Primarily one-way	4-32 kbit/s	< 1 s for playback < 2 s for record	< 1 ms	< 3% PLR	
Audio	High quality streaming audio	Primarily one-way	16-128 kbit/s (Note 3)	< 10 s	<< 1 ms	< 1% PLR	
Video	Videophone	Two-way	16-384 kbit/s	< 150 ms preferred (Note 4) <400 ms limit		< 1% PLR	Lip-synch: < 80 ms
Video	One-way	One-way	16-384 kbit/s	< 10 s		< 1% PLR	

NOTE 1 – Assumes adequate echo control.

NOTE 2 – Exact values depend on specific codec, but assumes use of a packet loss concealment algorithm to minimise effect of packet loss.

NOTE 3 – Quality is very dependent on codec type and bit-rate.

NOTE 4 – These values are to be considered as long-term target values which may not be met by current technology.

One-way delay

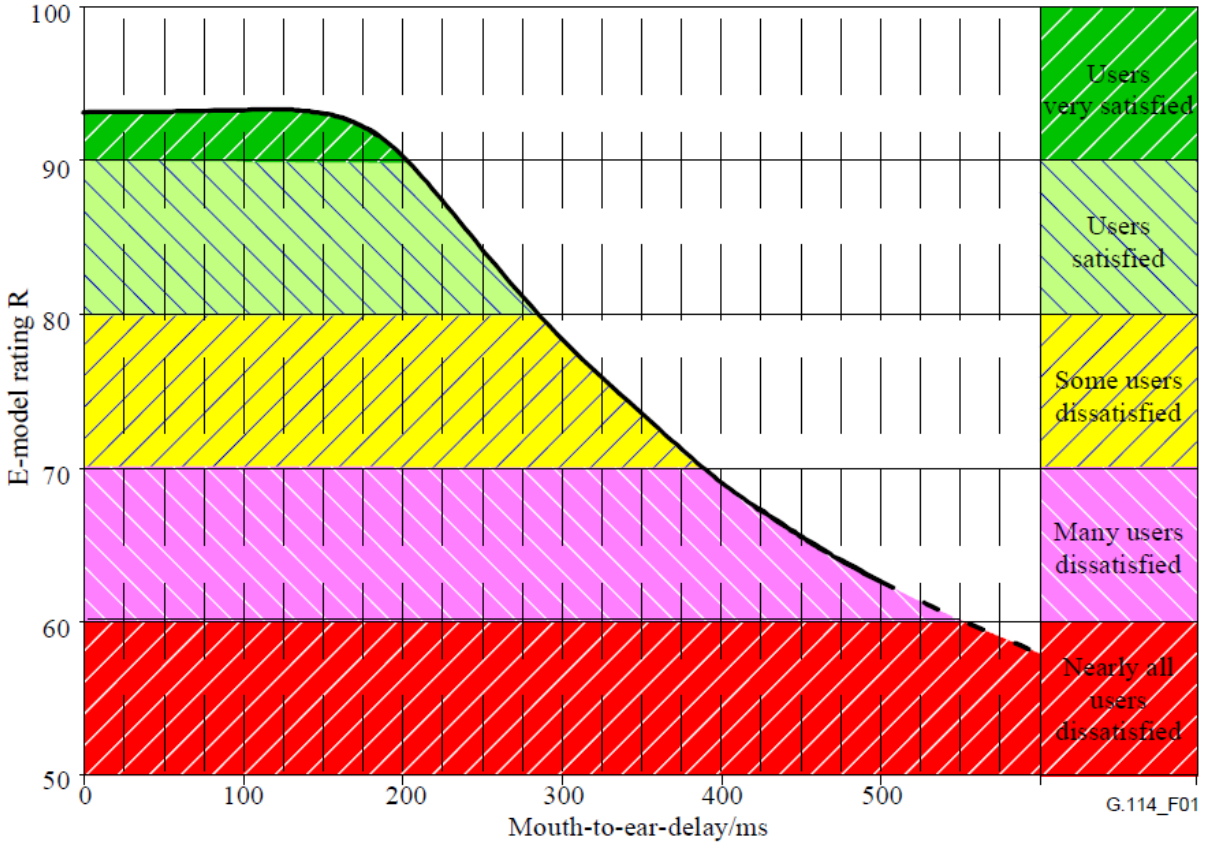
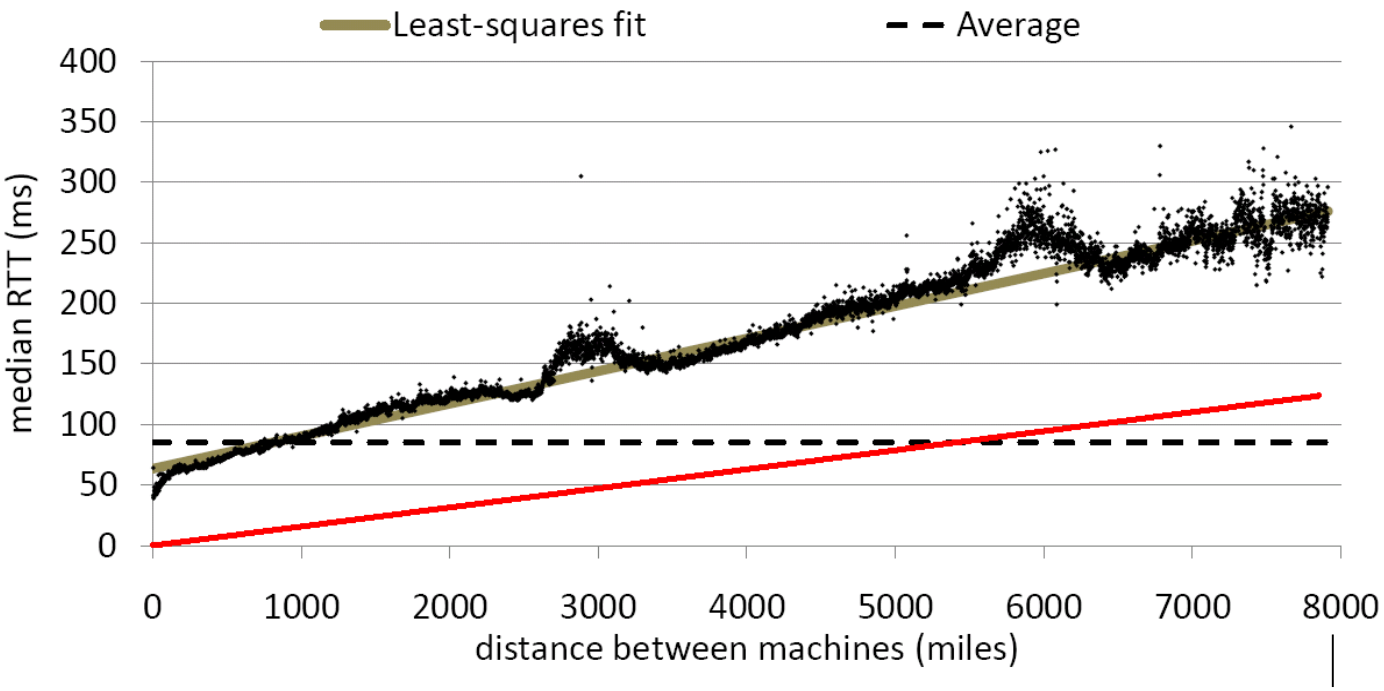


Figure 1/G.114 – Determination of the effects of absolute delay by the E-model

RTT, network delay



RTT (=ida y vuelta) en función de distancia
Medido para jugadores de Halo-3.
En rojo, retardo de transmisión (RTT) mínimo
Agarwal, Lorch [Sigcomm09]

12874 km

#2: ¿cumple parámetros de calidad?

#2= buffering, acumula algunos datos antes de comenzar reproducción

- **One-way delay**
  - **<150 ms preferred**
  - **<400 ms limit**
- **One-way delay =**

coding delay + reception delay + buffering delay

  - **Reception delay = network delay + transport-layer delay**

#2: ¿transport-layer delay?

#2= buffering, acumula algunos datos antes de comenzar reproducción

Suponemos TCP, ¿transport-layer delay?

- TCP puede empezar lento (control de congestión)
- TCP incorpora retardos si se pierden paquetes
  - ✓ Retardo por pedir retransmisión: ~1 RTT
  - ✓ Bajada de velocidad por mecanismo de control de congestión
  - Recordar que los problemas de retardos se agravan con la distancia
- Aplicación tolera '< 3% packet loss ratio', pero no retardos grandes
- Transport-layer delay en UDP = 0
- Solución #3: utilizar UDP
  - Nota: TCP puede funcionar, hay quien lo utiliza para comunicaciones interactivas

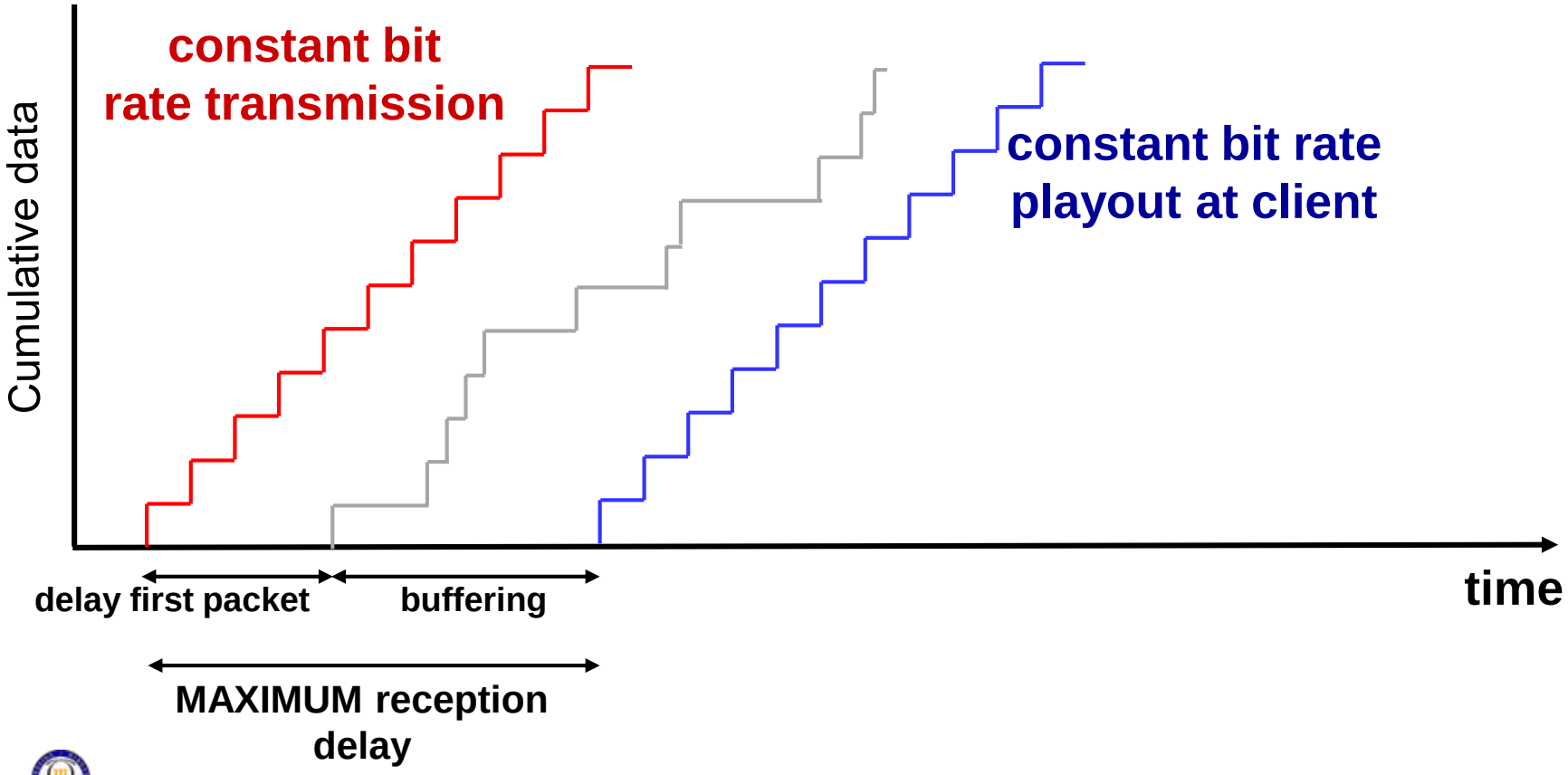
#3: ¿cumple parámetros de calidad?

#3= UDP con buffering,

- One-way delay
  - <150 ms preferred
  - <400 ms limit
- Buffering incrementa one-way delay ☹
- ¿Cuánto buffering?

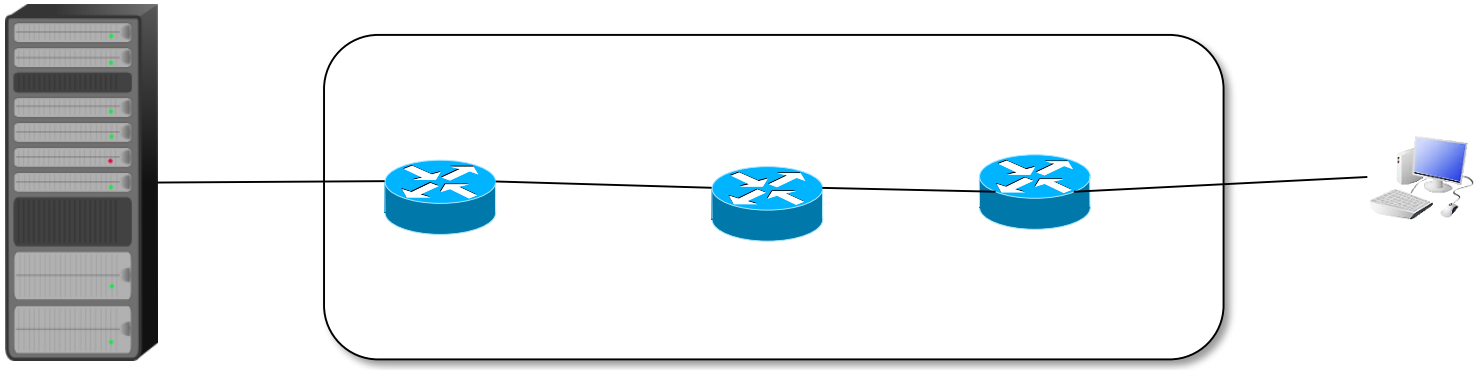
Dimensionamiento del buffer

- Buffer delay: se establece para el primer paquete
- Idealmente, debería ser suficiente para compensar el retardo máximo de la red (buffering >= retardo máximo – retardo primer paquete)



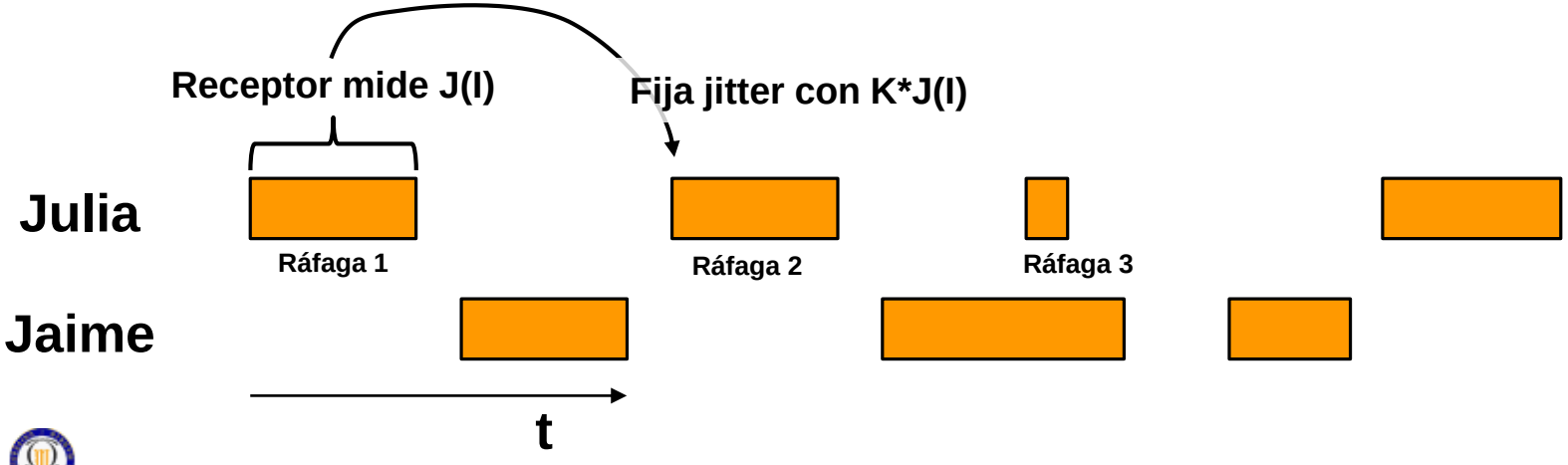
Dimensionamiento del buffer

- ¿Puedo conocer retardo one-way máximo?
  - ¿De qué depende?
- Pregunta previa: ¿puedo MEDIR el retardo one-way?
  - Requiere relojes sincronizados en FRECUENCIA y OFFSET
  - ¿Alguna estimación del retardo one-way?



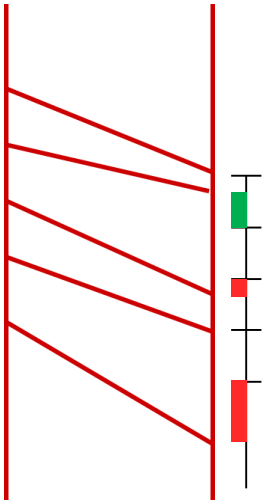
Dimensionar buffer

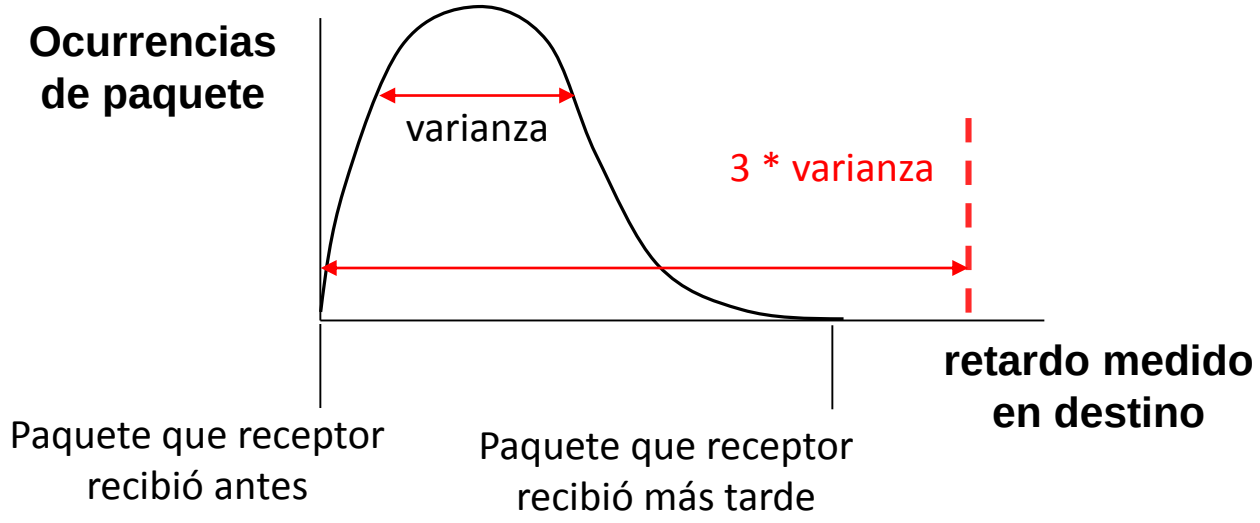
- Para un streaming ininterrumpido, el buffering se determina una vez, al principio de la comunicación
  - Ej: audio de una película
- Pero para una comunicación interactiva se eliminan los silencios y para cada periodo de habla activa (*talk spurt*) se puede determinar un tamaño de buffer distinto
  - Utilizando K * J(I)



Dimensionamiento del buffer

- **Supongamos que no conozco retardo máximo, ni puedo medir retardo**
  - **(restas de tiempos medidos en relojes que no sincronizan OFFSET y FRECUENCIA no tienen sentido)**
- **Heurístico para estimar un valor de retardo que NO se va a exceder (equivalente al retardo máximo):**
  - **Tomar derivada del retardo (= variación en el retardo), si la pudiera medir**
  - **Suponer que el retardo es gaussiano => muy baja probabilidad de ocurrencia de eventos un número K (bajo) de 'varianzas del retardo' respecto a la media**
    - ✓ **Baja probabilidad de agotar buffer**
  - **La próxima vez que tenga que dimensionar buffer, utilizar este valor**





#3 y pérdidas de paquetes

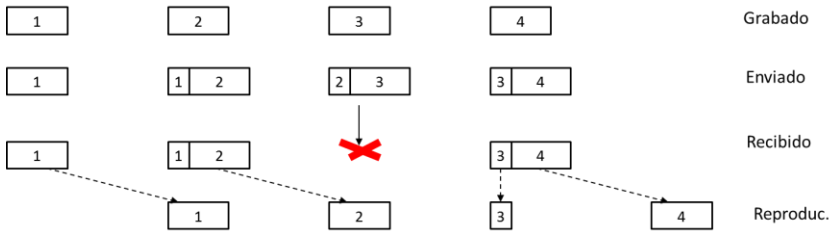
#3= UDP con buffering,

- ¿Puedo perder paquetes en #3 por 'problemas de control de flujo'?
  - TCP integra control de flujo, pero UDP no
- Tratamiento de pérdidas de paquetes en la red
  - TCP integra entrega fiable
  - UDP no gestiona pérdidas de forma automática, es responsabilidad de la aplicación

#3 y pérdidas de paquetes

#3= UDP con buffering,

- **Estrategia frente a pérdidas de paquetes en la red**
  - **Pérdidas puntuales**
    - ✓ **La aplicación pide retransmisión**
      - ➢ Si el retardo es muy bajo
    - ✓ **La fuente introduce redundancia**
      - ➢ Enviar más datos
    - ✓ **'Maquillar' pérdidas puntuales**
      - ➢ Interpolación de paquetes adyacentes, ruido blanco…



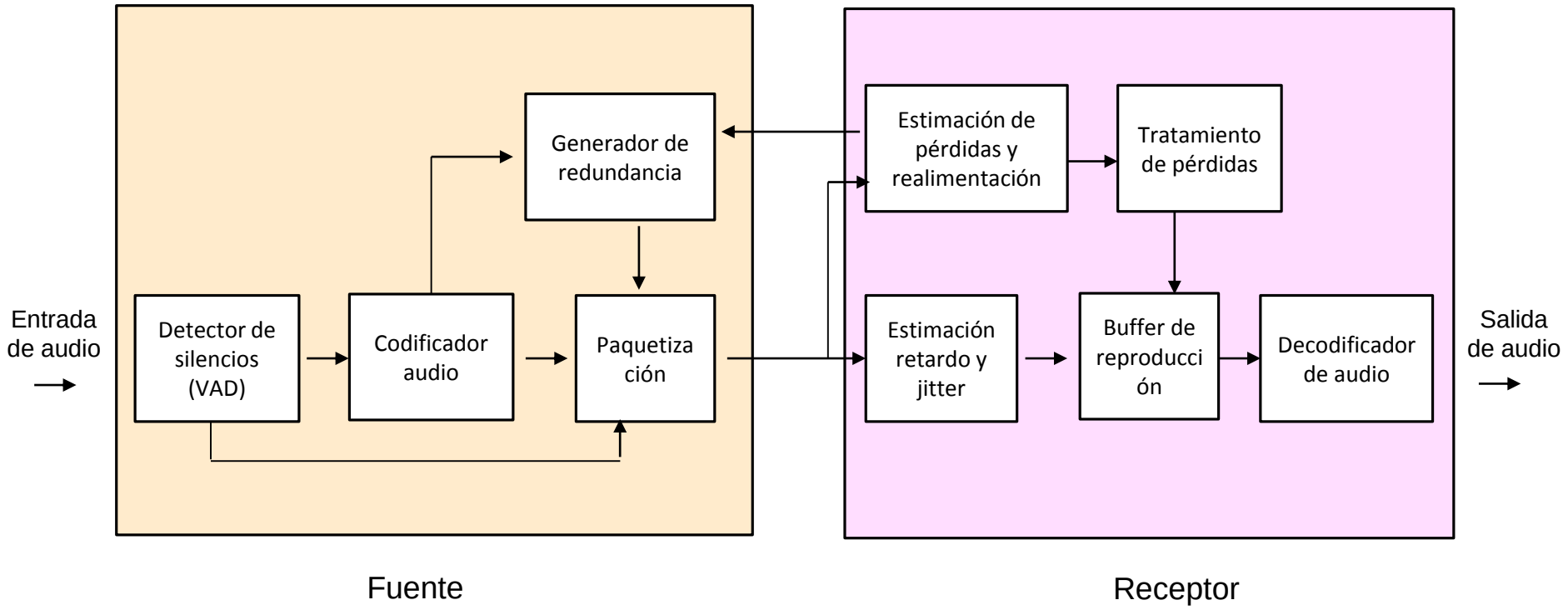
  - **Pérdidas continuadas: Reducir calidad ante pérdidas continuadas (posible congestión)**
    - ✓ **Receptor detecta pérdidas continuadas**
    - ✓ **Receptor transmite evento a fuente**
    - ✓ **Fuente cambia codec**
- **Necesitamos NUMERAR LOS PAQUETES para detectar pérdidas**
- **UDP no numera los paquetes**

#3 y temporización

#3= UDP con buffering,

- **A veces el tiempo de reproducción está implícito en los datos**
  - **Ejemplo, al 'echar datos seguidos' a la tarjeta de sonido, la tarjeta de sonido sabe qué hacer con ellos**
- **En otros casos, el tiempo de reproducción no está implícito en el flujo de datos**
  - **Audio con silencios: ¿cuándo termina el silencio?**
  - **Sincronización de un flujo de video y de subtítulos**
  - **Hace falta MARCAS de TIEMPO**
    - ✓ **Ni UDP ni TCP incluyen esta información**

Arquitectura de un sistema de VoIP



RTP

Departamento de
INGENIERIA
TELEMATICA
www.it.uc3m.es

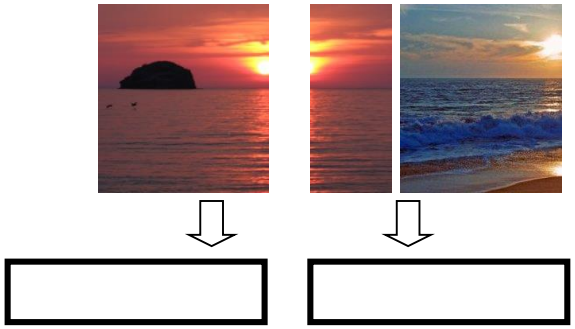
Introducción RTP

- La mayoría de las aplicaciones multimedia en red usan números de secuencia y marcas de tiempo
  - RTP ofrece este servicio de forma estandarizada
- Diseñado para poder usarse en una grandísima variedad de escenarios
  - Desde audioconferencias punto a punto a distribución de contenidos multimedia a miles de usuarios
  - Permite transportar formatos comunes PCM, GSM, MP3, … para sonido y MPEG, H.263, H.264, etc., para video
    - ✓ También transporta formatos propietarios
- Es el estándar utilizado para transporte de datos en muchas arquitecturas: SIP, H.323, RTSP…
- RTP
  - NO gestiona QoS (no tiene que ver con la infraestructura de red)
  - NO garantiza entrega fiable ni ordenada
- Va acompañado del intercambio de información de control: RTCP

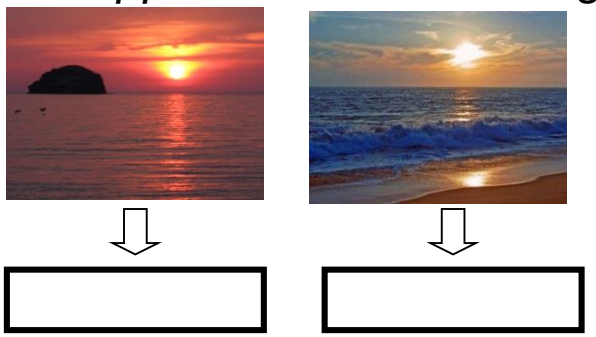
Filosofía RTP

- **Para datos multimedia, nadie mejor que la aplicación para gestionar el empaquetado de los datos**
  - **Resultados mejoran si tenemos en cuenta formato del contenido, y de los requisitos de calidad**
    - ✓ **Application level framing: La aplicación aprovecha conocimiento del formato del contenido para establecer los bloques en los que se envía la información**

Sin *application level framing*



Con *application level framing*



- ✓ **Un mensaje (nivel de aplicación) debe empaquetarse en un paquete (nivel de red)**
- ✓ **Requisitos de calidad: la aplicación sabe qué hacer si se pierden paquetes o llegan tarde**

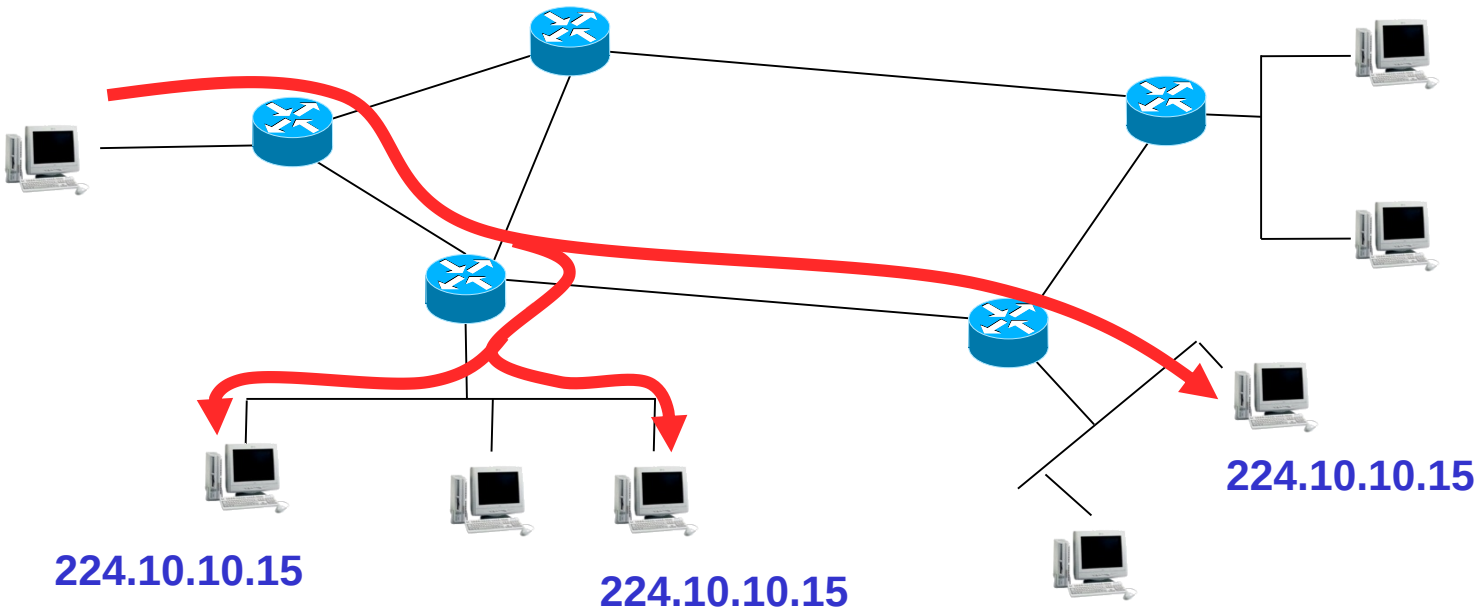
- **TCP ofrece un servicio independiente del contenido y que no tiene en cuenta los requisitos de calidad**

Filosofía RTP

- RTP funciona **extremo a extremo**: los nodos intermedios no interpretan RTP
- Necesita de un protocolo de transporte por debajo para multiplexación
  - RTP viaja preferentemente encima de UDP
- Conceptualmente, se puede entender como un servicio de **transporte**
  - Aunque la frontera con aplicación no está clara
- Respecto a su **implementación**, se suele integrar como parte del procesamiento a nivel de la aplicación que lo usa, en espacio de usuario, por encima del interfaz de sockets
  - Lo implementa el programador de la aplicación, o éste utiliza librerías que compila con su programa
  - No es un servicio del sistema operativo

Repaso: multicast en IP

- Direcciones entre 224.0.0.0 y 239.255.255.255
  - Sólo se pueden utilizar como dirección destino
- ¿Qué tiene que hacer un nodo para enviar a un grupo multicast?
- ¿Qué tiene que hacer un nodo para recibir de un grupo multicast?



Sesión RTP

- Sesiones “ligeras”: no hay control estricto ni centralizado sobre quién está en la sesión
  - Se ajusta bien al modelo multicast de IP
- *Sesión RTP* es una asociación entre un conjunto de participantes que se comunican con RTP

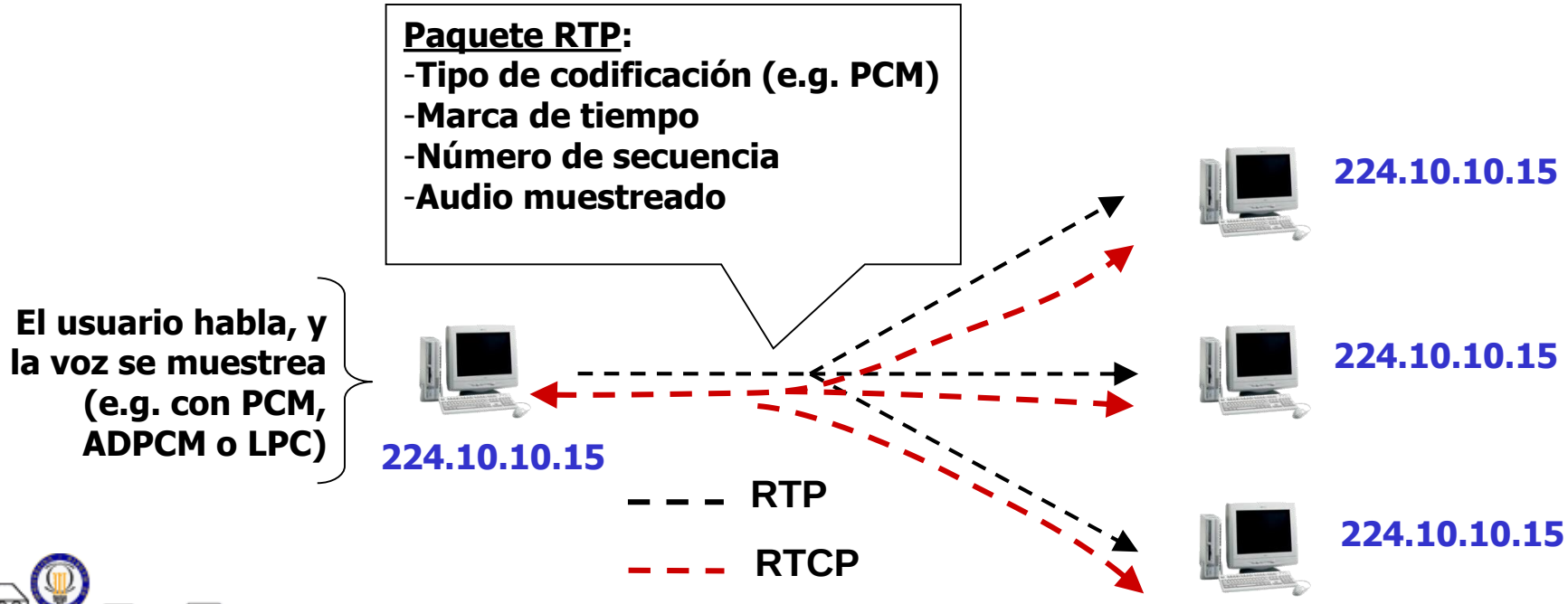
  Indica los parámetros de destino de los paquetes, y está definida por
  - Direcciones IP de los participantes
    - ✓ IP multicast
    - ✓ o lista de direcciones unicast
  - Puerto RTP: Puerto destino (UDP), en general un número par,
    - ✓ En general, lo utilizan TODOS los participantes (aunque podrían definirse puertos distintos para cada destino)
    - ✓ Puerto por defecto: `5004`
  - Puerto RTCP: Puerto destino (UDP) de los mensajes de control (RTCP), en general impar
    - ✓ En general, lo utilizan TODOS los participantes (aunque podrían definirse puertos distintos para cada destino)
    - ✓ En general, puerto RTCP = puerto RTP asociado + 1
- RTP no dice cómo establecer las sesiones, determinar los parámetros que las definen
  - Esto se hace a través de otros protocolos, ej. SIP

Sesión RTP

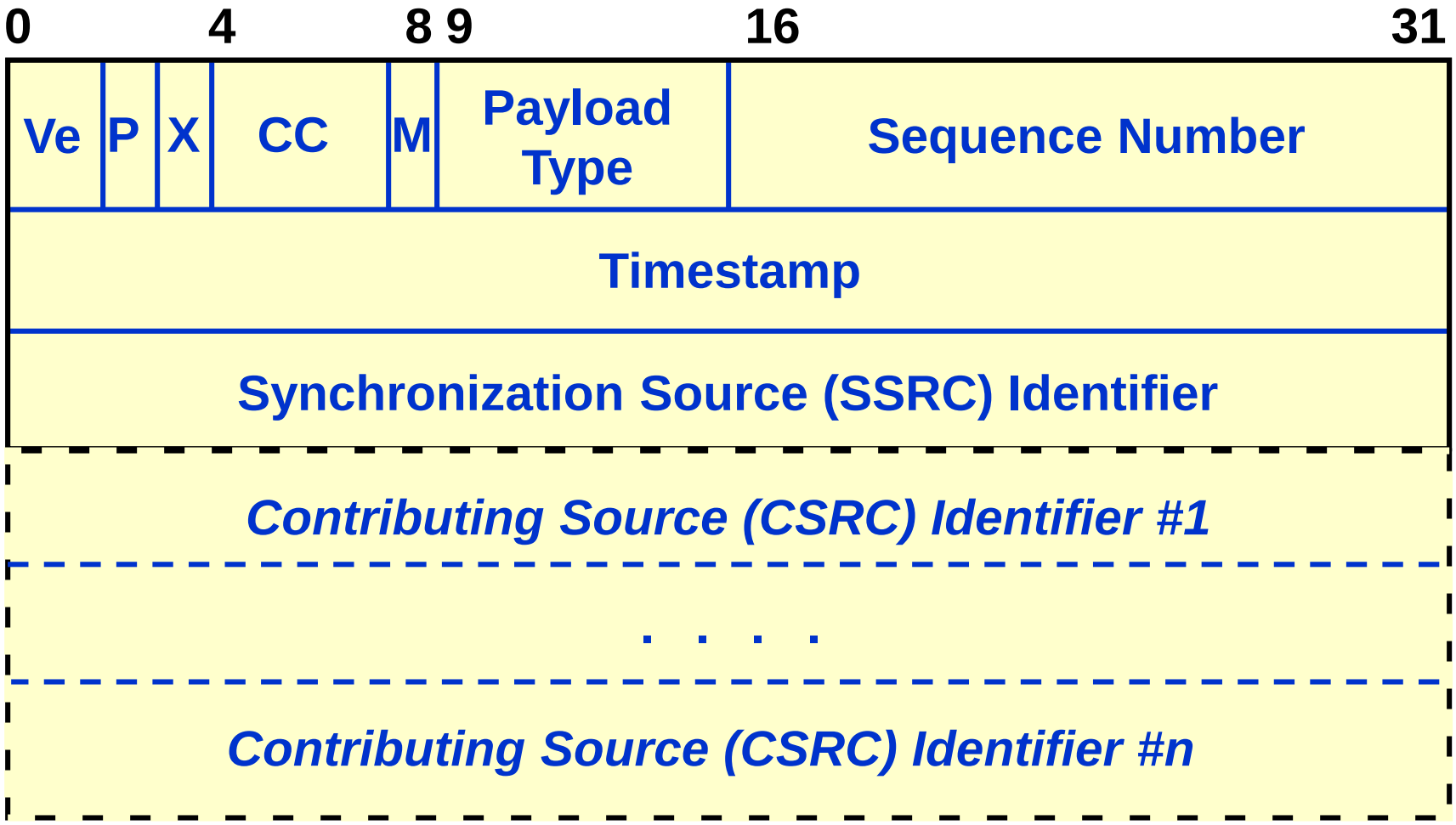
- Si un nodo participa en una sesión, es porque quiere recibir los datos (RTP) que se envían ahí
  - Además, alguno (o varios) de los que participan puede mandar datos (RTP)
  - Roles posibles: 'Receptor de datos', 'Emisor+Receptor de datos'
    - ✓ Los roles están asociados a un periodo de tiempo, delimitado por el envío de información RTCP
    - ✓ Los roles pueden cambiar dinámicamente (uno puede mandar un momento, y luego sólo recibir)
- Todos los nodos mandan y reciben información de control (RTCP)
  - También los que no mandan datos

Ejemplos de uso de RTP

- **Conferencia de audio multicast:**
  - 4 participantes
  - 1 sesión RTP: dirección *multicast* 224.10.10.15, puerto RTP 5004, puerto RTCP 5005
  - El audio se muestrea en cada equipo de usuario y se envía en paquetes RTP
  - Cada paquete RTP se envía en 1 paquete UDP, con dirección destino 224.10.10.15 y puerto destino 5004



Formato de Mensaje RTP



Contenido Multimedia

Profiles y payload types

Perfil (profiles): contextos en los que se define de forma más detallada cómo usar RTP

- **Ejemplos:**
  - **RTP/AVP (Audio Video Profile): Perfil genérico para conferencias de audio/vídeo con mínimo control (RFC3551: "RTP Profile for Audio and Video Conferences with Minimal Control")**
  - **RTP/I Para *whiteboards*, y medios interactivos**
  - **Para transporte de audio y vídeo con retransmisión, …**
- **Un perfil define**
  - **Números de payload, que tienen significados distintos en cada perfil (se 'reutiliza' el mismo número en distintos perfiles, con significados diferentes)**
  - **Cómo se empaqueta la información multimedia para cada tipo de payload (cómo se calcula el Timestamp, cuántos datos meter por paquete…)**
    - ✓ **También puede haber documentos específicos en los que se especifica el payload format**
  - **Incluso refinar el comportamiento de RTP/RTCP en ese perfil**
    - ✓ **Cambiar intervalo de generación de paquetes RTCP**
    - ✓ **Añadir nuevos tipos de paquete RTCP**
    - ✓ **Definir extensiones para RTP**
    - ✓ **...**

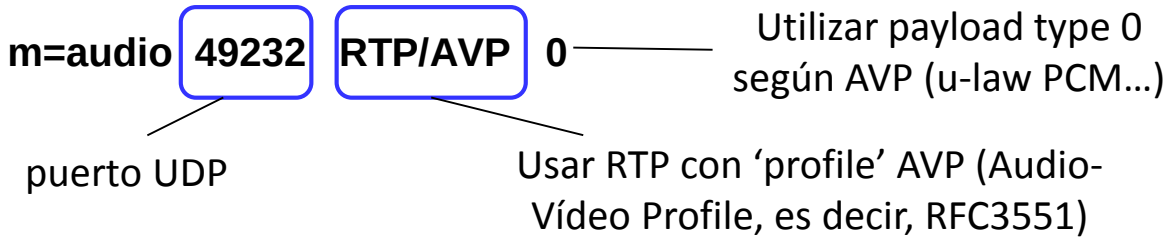
Algunos documentos de “profiles”

RTP Profile for Audio and Video Conferences with Minimal Control (RFC 3551)
The Secure Real-time Transport Protocol (SRTP) (RFC 3711)
Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF) (RFC 5124)
Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF) (RFC4585)

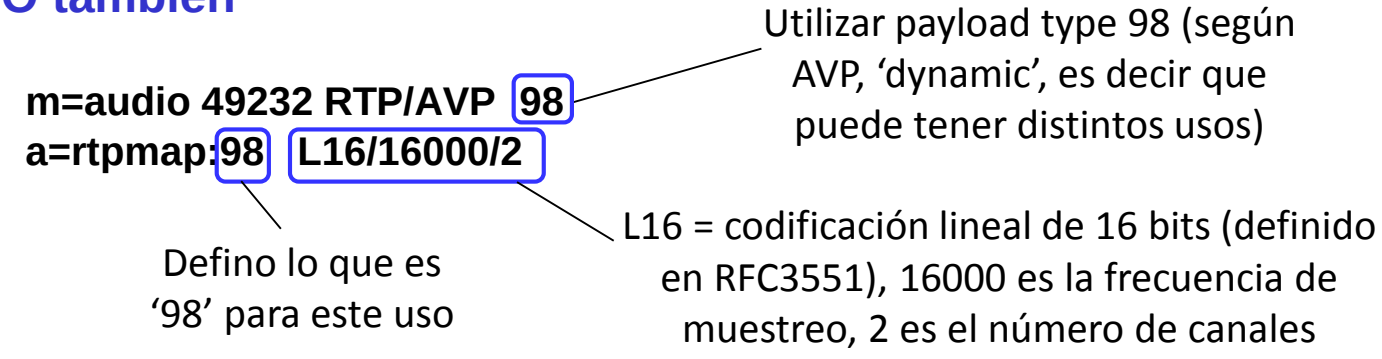
RTP Payload Format for JPEG-compressed Video (RFC 2035) obsoleted by RFC 2435
RTP Payload format for H.261 video streams (RFC 2032)
RTP Payload Format for MPEG1/MPEG2 Video (RFC 2038) obsoleted by RFC 2250
RTP Payload Format of Sun's CellB Video Encoding (RFC 2029)
RTP Payload Format for H.263 Video Streams (RFC 2190)
RTP Payload for Redundant Audio Data (RFC 2198)
RTP Payload Format for MPEG1/MPEG2 Video (RFC 2250)
RTP Payload Format for Bundled MPEG (RFC 2343)
RTP Payload Format for the 1998 Version of ITU-T Rec. H.263 Video (H.263+) (RFC 2429)
RTP Payload Format for BT.656 Video Encoding (RFC 2431)
RTP Payload Format for JPEG-compressed Video (RFC 2435)
An RTP Payload Format for Generic Forward Error Correction (RFC 2733)
RTP Payload for Text Conversation (RFC 2793)
RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals (RFC 2833)
RTP Payload Format for Real-Time Pointers (RFC 2862)
RTP payload format for MPEG-4 Audio/Visual streams (RFC 3016)
RTP Payload Format for ITU-T Recommendation G.722.1 (RFC 3047)
A More Loss-Tolerant RTP Payload Format for MP3 Audio (RFC 3119) (37796 bytes)
RTP Payload Format for DV Format Video (RFC 3189) (25991 bytes)
RTP Payload Format for 12-bit DAT, 20- and 24-bit Linear Sampled Audio (RFC 3190) (34977 bytes)
RTP payload format and file storage format for the Adoptive Multi-Rate (AMR) and Adaptive Multi-RTP Payload for Comfort Noise (RFC 3389) (17018 bytes)
…

Negociación de perfiles y 'payload types'

- **Todas las aplicaciones que pertenecen a una misma sesión RTP deben utilizar el mismo perfil**
  - **Pero NO hay un 'identificador de perfil' que viaje con los paquetes**
- **Se pueden utilizar distintos protocolos para negociar el perfil (nota: 'fuera' de RTP)**
- **SDP (Session Description Protocol, RFC 4556), define un formato en el que se especifican los parámetros de la sesión multimedia**

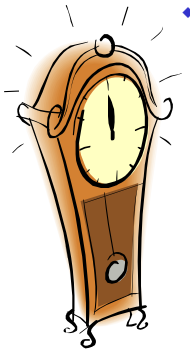


- **O también**



Formato de Mensaje RTP

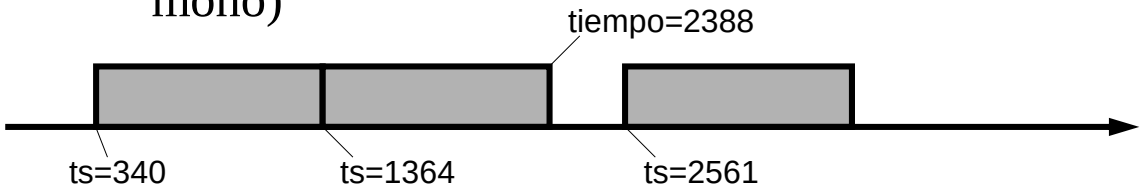
- **Marca de tiempo (*timestamp*):**
  - **Permite sincronización de medios**
  - **Cálculo de *jitter* de paquete (útil para RTCP)**
  - **… ¿con qué formato? (con qué precisión…)**
    - ✓ **Ej: poner texto de subtítulos en instante 19.025 s, siguiente en 27.145 s**
    - ✓ **Empezar a reproducir paquete de audio en 0 s, siguiente en 1/3 de segundo, luego 2/3 de segundo…**
      - 0.33333333333333 s ?
  - **Diferentes formatos de marca de tiempo en función de perfil y payload**
  - **No debe requerir sincronización entre relojes: sólo indica valores de tiempo relativos entre paquetes**
    - ✓ **Difícil requerir sincronización de *offset* entre relojes (razonable pedir una frecuencia similar)**



Formato de Mensaje RTP

- **Marca de tiempo (*timestamp*): 32 bits**
  - **Como es dependiente del perfil, consideremos ejemplo de 'audio' en RFC3551**
    - ✓ ***Timestamp* indica el instante de reproducción del primer octeto de datos contenido en el paquete RTP**
    - ✓ **Emplea el reloj (virtual) de muestreo del audio (NO reloj del sistema operativo, ni CPU)**
      - No tiene sentido pedir a la tarjeta de sonido un dato "en medio" de un periodo de muestreo. Tampoco iniciar una reproducción ahí (después de un silencio, por ejemplo)

      - Ejemplo: en audio sin comprimir el tiempo pasa en 1 unidad en **cada periodo de muestreo**
        - Suponer paquete de 2048 bytes, 2 bytes por muestra (16 bits mono)

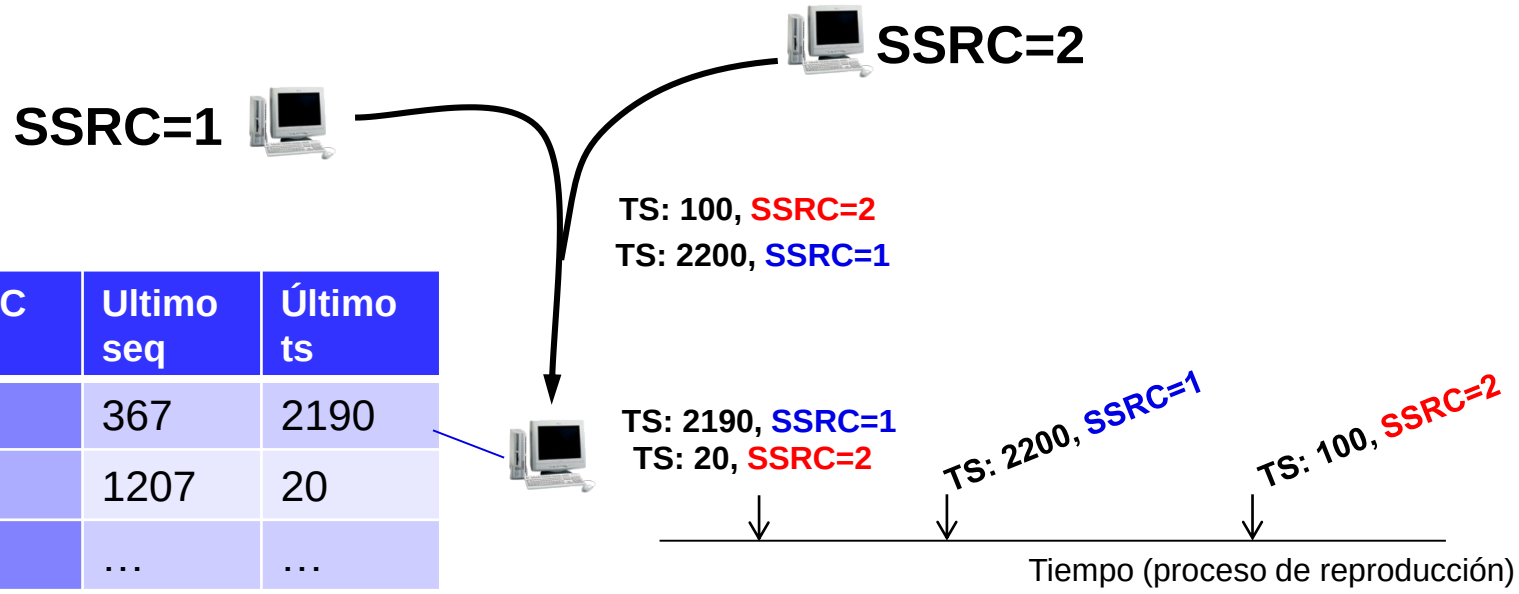


Formato de Mensaje RTP

- **Marca de tiempo (*timestamp*): 32 bits**
  - **Valor inicial es aleatorio (como el número de secuencia)**
  - **Ejemplos de otros perfiles**
    - ✓ **En video, si hay varios paquetes para transportar un mismo *frame* (porque no quepan en un único paquete), llevan igual timestamp**
    - ✓ **MPEG marca de forma no monótona creciente (interpolados se envían después)**

Formato de Mensaje RTP

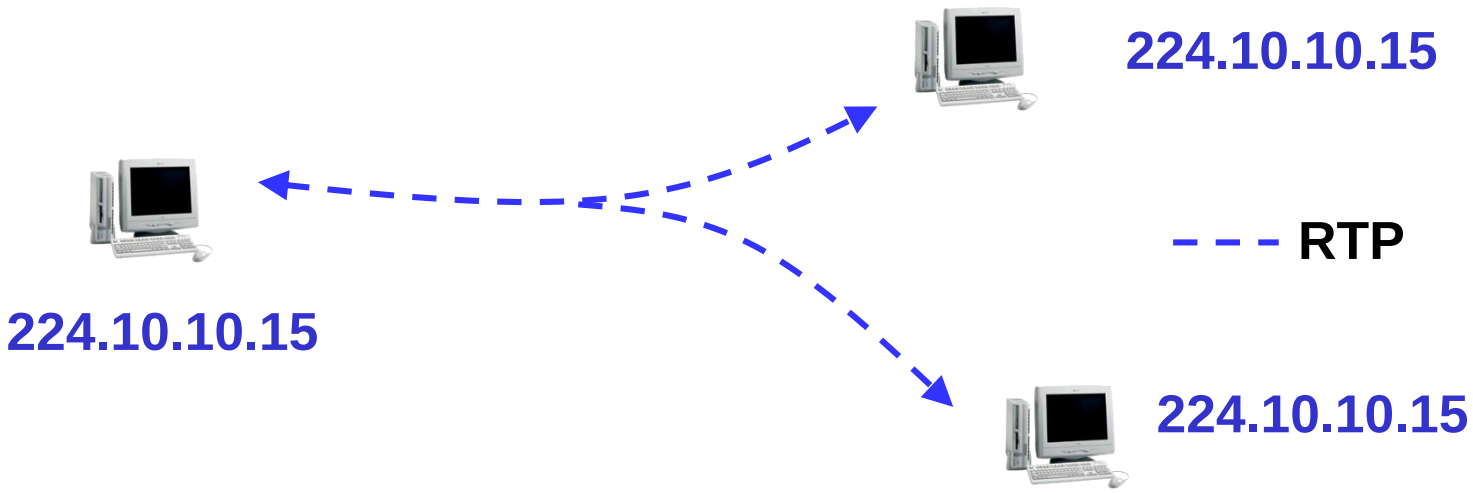
- **SSRC (*Synchronization Source Identifier*): 32 bits**
  - Generado aleatoriamente por la fuente
  - Identificador único a nivel RTP de una fuente para una sesión
  - Utilizado para asociar estado RTP a cada fuente
    - ✓ mantener la temporización de la reproducción de cada fuente
    - ✓ números de secuencia
    - ✓ …



SSRC	Ultimo seq	Último ts
1	367	2190
2	1207	20
…	…	…

“Sesión ligera” en RTP

- **Control: Las sesiones RTP son “muy” ligeras, como es el control de IP multicast**
  - **Cualquiera puede unirse a una sesión en cualquier momento**
    - ✓ **Si queremos restringir, puede hacerse cifrando y distribuyendo las claves con otros protocolos de nivel superior**
  - **El emisor no sabe cuántos destinatarios hay**
    - ✓ **¡Puede ocurrir que no haya ninguno!**
- **Parece conveniente dar algún tipo de realimentación a los participantes – sobre todo a los emisores => RTCP**



RTCP (Real Time Control Protocol)

Funciones

- **Monitorización de la congestión y del QoS**
  - Se envían informes sobre lo recibido de cada participante en una sesión
  - Permite medir *jitter*, y RTT, y calcular tiempo de buffering
- **Identificación**
  - Identificador de la fuente mediante una cadena de caracteres
  - Puede utilizarse para asociar flujos de diferentes sesiones
  - También permite enviar el nombre del usuario y otros parámetros textuales de tipo informativo
- **Control de sesión mínimo**
  - Abandono de sesión
  - Conocer el número de participantes

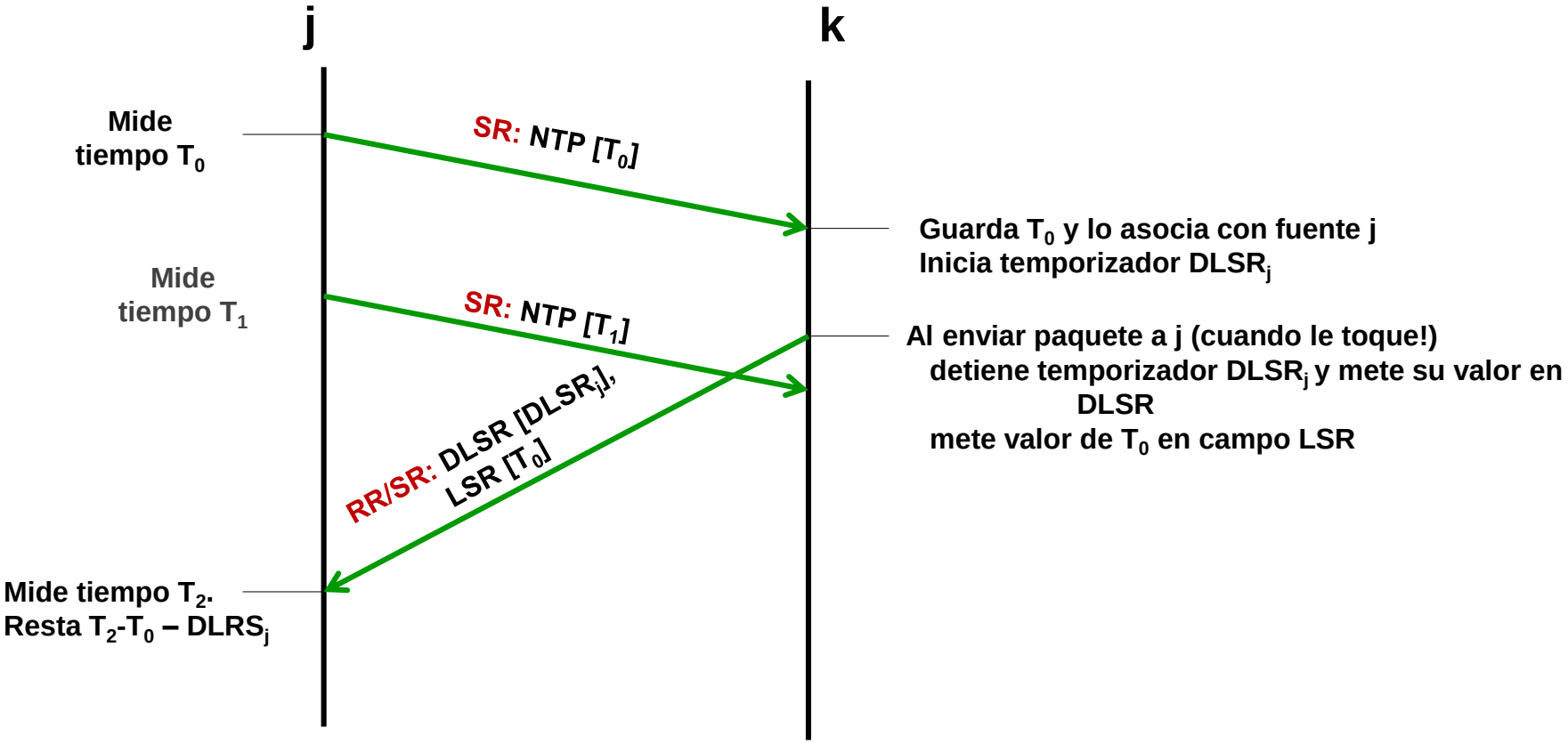
Paquetes RTCP

Cinco tipos

- **Sender Report (SR)**
  - Indica lo que ha enviado el nodo (y deberían haber recibido los demás) + lo que ha recibido el nodo
- **Receiver Report (RR)**
  - Informa sobre lo que ha recibido el nodo
- **Source Description (SDES)**
  - Descripción de la fuente: CNAME (*canonical name*), NAME, EMAIL, PHONE, LOC, TOOL, ...
- **BYE**
  - Opcional, indica que un equipo abandona la sesión, y la razón por la que la abandona
- **Específico de aplicación**
  - Opcional, permite añadir funcionalidades sin necesidad de definir nuevos paquetes en el protocolo

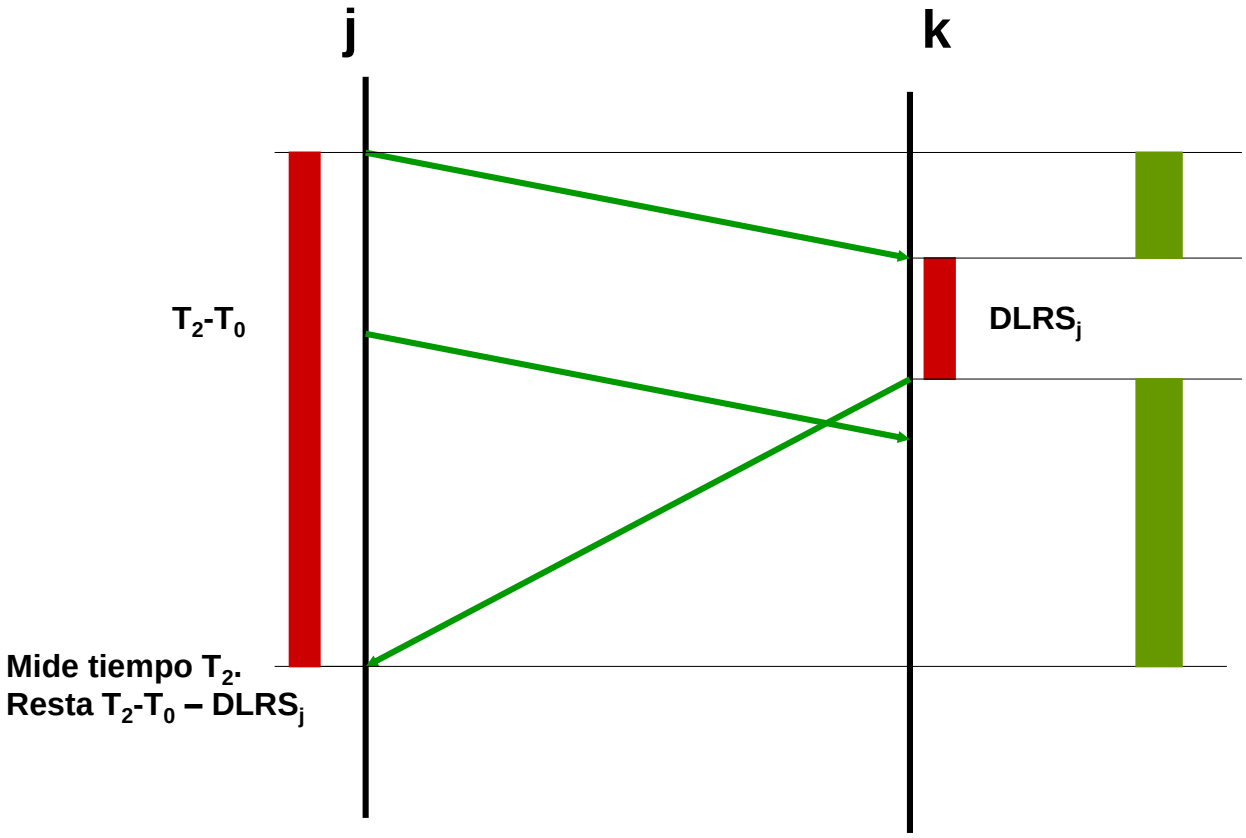
Cálculo del *Round Trip Time*

- **Objetivo: Mínimo número de mensajes para control**



- 'j' es un 'sender', 'k' puede ser cualquier cosa
  - 'k' manda información relevante en Rec Report Block
- Cada emisor recibe estimación de RTT con cada uno de los miembros de la sesión

Cálculo del *Round Trip Time*



- ‘j’ no tiene que almacenar estado
- ‘k’ sólo guarda por emisor RTP:
  - valor del campo NTP del último paquete recibido,
  - tiempo en el que recibió el último paquete

RTCP y Round Trip

- **Cálculo del Round Trip entre fuente j y k**
  - **j envía mensaje RTCP SR a k**
    - ✓ **instante T_0 codificado en NTP (64 bits)**
  - **k envía un tiempo después mensaje RTCP RR a j**
    - ✓ **Delay since Last SR ($DLSR_j$): tiempo desde que se recibió el mensaje de j hasta que k envía de vuelta a j**
    - ✓ **Time of Last Sender Report (LSR): tiempo de generación del primer mensaje en j según j (ahora en otro formato: 32 bits centrales del NTP)**
  - **Round trip: j recibe paquete de k en T**
    - ✓ **Round trip = T-LSR - DLSR**

Estado en los receptores

- **Cada receptor debe mantener estado PARA CADA FUENTE de la que ha recibido 'recientemente'**
  - **SSRC identifica la fuente,**
  - **RTP timestamp del último paquete recibido, asociado a un tiempo 'local' en el que se debe haber reproducido**
    - ✓ **Si no, no sabrá cuándo tiene que reproducir los contenidos de los paquetes de datos**
  - **Número de secuencia del último paquete recibido y del último paquete enviado antes de enviar el último mensaje SR o RR**
    - ✓ **Para poder calcular 'fraction lost', y saber si el siguiente paquete esperado se ha perdido o no…**
  - **DLSR asociado al último mensaje RTCP recibido de la fuente**
  - **Tiempo NTP recibido en el último mensaje RTCP recibido de la fuente**
  - **…**

Puertos en paquetes UDP

- RFC 3550 hace referencias sobre el puerto de destino de un paquete RTP o RTCP
  - Ej: dice que por defecto para RTP sea 5004,
  - En cualquier caso recomienda que el puerto para RTP sea par, y el de RTCP = RTP+1
  - El puerto origen podría ser cualquiera
- Sobre el puerto origen, RFC4961 sugiere que sea igual que el destino (tanto para RTP como para RTCP)
  - Ej: si capturáramos un paquete "en tránsito":
    - ✓ Dir orig: 163.117.140.180, puerto orig: 5004, dir dest: 225.10.0.3, puerto dest: 5004